РЕФЕРАТ

ВЕБ-ПРИЛОЖЕНИЕ ДЛЯ СИНТЕЗА, ХРАНЕНИЯ И РАСПРОСТРАНЕНИЯ АУДИОКНИГ КНИГ НА БАЗЕ SPRING FRAMEWORK: дипломный проект / В. В. Гринчик. – Минск: БГУИР, 2021, – п. з. – 107 с., чертежей (плакатов) – 6 л. формата А1.

Целью данного дипломного проекта является создание веб-приложения для синтеза, хранения и распространения аудиокниг. Актуальность данной работы обусловлена ростом популярности аудиокниг и современными тенденциями развития информационного общества, в связи с которыми возникла необходимость в дешевом и быстром способе создания, хранения и распространения аудиокниг.

В первом разделе проводится анализ литературных источников, по итогам которого была изучена история и тенденции роста популярности аудиокниг, особенности проектирования программных систем хранения и распространения аудиокниг, а также принципы и модели работы технологии синтеза речи. Кроме того, в этом разделе проводится анализ существующих аналогов проектируемого приложения, по итогам которого были выявлены и учтены достоинства и недостатки данных программных продуктов. На основе проведённого анализа выдвигаются общие требования к созданию веб-приложения.

Во втором разделе проводится моделирование предметной области, а также разработка функциональных требований.

Третий раздел посвящён разработке архитектуры веб-приложения и модели базы данных. Кроме того, в третьем разделе описывается разработка некоторых алгоритмов данного веб-приложения.

В четвёртом разделе рассматриваются технологии, используемые в приложении, а также описываются основные компоненты разрабатываемого приложения.

В пятом разделе содержится информация о тестировании разработанного приложения на соответствие функциональным требованиям.

Шестой раздел содержит руководство по установке и использованию разработанного приложения.

В седьмом разделе приведено технико-экономическое обоснование разработки и использования приложения.

Заключение содержит краткие выводы по дипломному проекту.

Дипломная работа прошла проверку в системе «Антиплагиат». Уникальность данного проекта составляет 97,44 %.

СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ И СОКРАЩЕНИЯ

В настоящей пояснительной записке применяются следующие определения и сокращения.

Internet – всемирная система объединённых компьютерных сетей для хранения и передачи информации.

Никнейм – псевдоним, используемый пользователем в сети Internet.

ОС – Операционная Система – комплекс взаимосвязанных программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем.

СУБД – Система Управления Базами Данных – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Фреймворк – программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

Пагинация – в веб-приложениях под пагинацией понимают постраничный вывод с показом ограниченной части информации на одной веб-странице и возможностью переключения между страницами.

ПО – Программное Обеспечение.

MVC – Model-View-Controller (Модель-Представление-Контроллер) – архитектурный паттерн, подразумевающий разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента.

ВВЕДЕНИЕ

Аудиокни́га (от лат. audio «слушать») — озвученное литературное произведение. Аудиокниги могут быть как развлекательными, так и просветительскими или образовательными. Сюда можно отнести аудиолитературу для инвалидов, слепых и людей с нарушенным зрением, начитанные сказки для детей младшего возраста, аудиокурсы иностранных языков, и тому подобное. В современном мире аудиокниги обретают всё большую популярность. Это обусловлено многими факторами.

К таким факторам можно отнести удобство прослушивания аудиокниг в ситуациях, не требующих серьёзной концентрации внимания и в то же время по различным причинам не позволяющих читать обычные книги. Например, поездки в общественном транспорте, приготовление пищи, уборка, спортивные тренировки и тому подобное. В подобных ситуациях удобство аудиокниг в первую очередь связано с тем, что они не занимают руки, не требуют использовать зрение, и даже слух продолжает воспринимать информацию, не связанную с аудиокнигой.

Ещё одним важным фактором является то, что прослушивания аудиокниг позволяет снять часть нагрузки со зрительных органов, а это, в свою очередь, положительно сказывается на их здоровье. Особенно важен данный фактор для работников сферы информационных технологий, которые много времени проводят, смотря в монитор компьютера, чем серьёзно напрягают зрение.

Также необходимо обратить внимание на то, что аудиокниги позволяют получить доступ к текстовой информации инвалидам и людям с нарушенным зрением. Что оказывает положительное влияние на их социально-психологическую адаптацию в обществе, и улучшает качество жизни.

На данный момент создание аудиокниги длительный и довольно дорогой процесс. И в связи с этим количество ежегодно выпускаемых аудиокниг значительно меньше количества печатных изданий. На текущий момент у данной проблемы есть два решения: заказать профессиональную озвучку книги по высокой цене или установить на свое устройство специальную программу для озвучивания книги в реальном времени. У каждого из решений есть свои недостатки. Разрабатываемый проект должен стать альтернативным решением, позволяющим не только создавать, но и хранить аудиокниги, а также распространять их в сети интернет.

Целью данного дипломного проекта является создание веб-приложения для синтеза, хранения и распространения аудиокниг. Актуальность темы обусловлена ростом популярности аудиокниг и современными тенденциями развития информационного общества, в связи с которыми возникла необходимость в дешевом и быстром способе создания, хранения и распространения аудиокниг.

1 АНАЛИЗ ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ, ПРОТОТИПОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРИЛОЖЕНИЮ.

Итоговый успех разработки и реализации программного проекта во многом определяется на этапе подготовки, во время которой необходимо как можно больше и тщательней определить нюансы и особенности проекта.

Первое предварительное условие, которое нужно выполнить перед конструированием, – ясное формулирование проблемы, которую система должна решать. Общая цель подготовки – снижение риска. Адекватное планирование позволяет исключить главные аспекты риска на самых ранних стадиях работы, чтобы основную часть проекта можно было выполнить максимально эффективно.

Главный факторы риска в создании ПО – неудачная выработка требований. Требования подробно описывают, что должна делать программная система. Внимание к требованиям помогает свести к минимуму изменения системы после начала разработки [1].

1.1 Анализ литературных источников

1.1.1 История и тенденции роста популярности аудиокниг

История появления и развития аудиокниг насчитывает много лет. Аудиокниги вызвали огромный интерес у людей во всем мире. Термин «аудиокнига» впервые появился в Германии в 1954 году. Это был год основания немецкой аудиобиблиотеки для слепых. В том же году «Немецкий граммофон» попытался записать инсценировку «Фауста». Эта запись, которая была выпущена в количестве 250 000 экземпляров, имела большой успех и считается началом основания и развития аудиокниги.

С 1963 года изобретение магнитофона дало возможность распространять и продавать записи на кассетах. Первыми слушателями кассет были слепые и с плохим зрением люди, которые благодаря особому качеству аудиокниг имели возможность пользоваться ими в любое время и в любом месте. Постепенно у аудиокниг появляется все больше и больше поклонников: любители литературы, домохозяйки, учащиеся школ и студенты, путешественники и т. д

В настоящее время популярность аудиокниг не угасла. А появление интернета открыло еще больше возможностей по сбыту аудиокниг, так как формат MP-3 позволяет это сделать без всяких проблем, экономя расходы на их производство, хранение, упаковку и пересылку.

Изменения информационной и общественной среды также вносят свой вклад в растущую в последнее время популярность аудиокниг. К таким изменениям можно отнести:

– визуальное перенапряжение ежедневным потоком информации (дорожные знаки, рекламные щиты, витрины и т. д.);

– работа за компьютером вызывающая соответствующее напряжение глаз;

– ежедневный поток информации из печатных изданий (газеты, журналы, специальная литература и т. д.);

– деятельность, связанная с общими мыслительными процессами и оставляющая место для дополнительной информации (вождение автомобиля, перемещение в общественном транспорте).

Из всего вышесказанного можно сделать вывод, что аудиокниги пользуются большой популярностью благодаря своей развлекательной функции, особенно среди людей, чья деятельность связана со зрительным напряжением. Аудиокниги помогают расслабиться, их можно слушать с закрытыми глазами, а также, благодаря гибкости их использования, в любом месте и в любое время [2].

На основе проведенного анализа тенденций роста популярность аудиокниг, можно сделать вывод как о текущем высоком спросе на аудиокниги, так и о том, что спрос будет продолжать расти. Что позволяет рассчитывать на высокую популярность проектируемого приложения, и как следствие получение значительной прибыли от разработки.

1.1.2 Особенности проектирования программных систем хранения и распространения аудиокниг

По своей технологии и организации программная система хранения и распространения аудиокниг наиболее близка к электронной библиотеке.

Первые шаги по созданию электронных библиотек (ЭБ) были сделаны за рубежом в начале 1980-х гг. В 1992 г. на конференции Национального научного фонда США было введено в оборот понятие «цифровая библиотека» в современном контексте.

В общем случае при проектировании ЭБ следует рассматривать два класса требований, которые можно назвать пользовательскими и общесистемными. Пользовательские требования определяют содержание фонда, его структуру, систему метаданных и функциональные возможности ЭБ. Общесистемные требования определяют общую структуру ЭБ, технологию функционирования ЭБ в рамках действующей организации с учетом ее задач и специфики, взаимодействия с другими организациями, порядок ее использования и администрирования.

Все информационное пространство ЭБ, доступное пользователю, должно быть представлено в виде совокупности самостоятельных объектов. В качестве таковых во многих случаях выступают электронные документы. Электронные объекты в общем случае могут представлять собой текстовые произведения, изображения, аудиофайлы, базы данных или их фрагменты,

словарные статьи, подписи под рисунками, отдельные имена и т. д. Организация информационного пространства как совокупности объектов и однозначная идентификация последних необходимы для обеспечения эффективной навигации и выполнения некоторых видов информационных поисков. Инструментом описания и идентификации выступают метаданные, в том числе библиографические записи, поскольку основную часть фонда будут составлять обычные документы.

Для представления документов в ЭБ могут использоваться разные форматы, в том числе:

– формат PDF;
– форматы DOC (DOCX), TXT;
– аудиоформаты, например, MP3;
– гипертекстовый язык разметки HTML;
– расширенный язык разметки текста XML.

Выбор одного или нескольких форматов для хранения определяется в рамках концепции ЭБ с учетом пользовательских и общесистемных требований [3].

Таким образом разрабатываемое в рамках данного дипломного проекта приложение может быть определено как электронная библиотека, хранящая документы в аудиоформате. А принципы, изложенные ранее, могут применятся при разработке технического задания к приложению.

1.1.3 Модели работы технологии синтеза речи

По сути любые системы синтеза аудиокниг основаны на технологии синтеза речи.

Все существующие в настоящее время методы синтеза человеческой речи основаны на использовании двух моделей — модели компилятивного синтеза и формантно-голосовой модели.

Модель компилятивного синтеза предполагает синтез речи путем конкатенации (составления) записанных образцов отдельных звуков, произнесенных диктором. При использовании этой модели составляется база данных звуковых фрагментов, из которых в дальнейшем будет синтезироваться речь [4]. Преимуществом данной модели является простота реализации. Недостатком требовательность к образцам звуков.

Формантно-голосовая модель основана на моделировании речевого тракта человека. При форматном методе, моделируется результат физиологических процессов образования речи: акустические характеристики речевой волны. Эта модель может быть реализована с применением нейронных сетей и допускает самообучение. К сожалению, ввиду сложности точного моделирования особенностей речевого тракта, а также учета интонационной модуляции речи формантно-голосовая модель обладает относительно низкой точностью синтезируемых звуков речи [5].

«Zvukogram» – это платный сервис, с возможностью бесплатного озвучивания небольшого пробного теста. Интересной особенность данного веб-приложения является то, что в нем используется сразу две модели синтеза речи, компилятивная и форматно-голосовая. На сайте доступно около 150 голосов, на 11 языках. В сервисе нет ограничений. Можно сразу озвучить большую статью и все это будет в одном файле.

Изначально каждому пользователю доступно озвучивание 5000 символов простым голосом. При регистрации на счет пользователя зачисляется еще 10000 символов. Один символ, озвученный премиум голосом, равен 5 простым символам. Стоимость одной тысячи символов один рубль.

В отличие от любых других приложений для озвучивания текста, «Zvukogram» позволяет озвучивать текст сразу несколькими голосами, в том числе на разных языках. Это позволяет озвучивать диалоги, и обучающие тексты, содержащие несколько языков.

При озвучивании приложение позволят добавлять в текст паузы и различные звуковые эффекты.

Преимущества:

– более 150 голосов для озвучивания на 11 различных языках;

– возможность скачать полученный аудио файл в нескольких форматах;

– возможность прослушать результат прямо на сайте;

– возможность при озвучивании построить диалог используя несколько голосов одновременно;

– добавление в тексты различных звуковых эффектов.

Недостатки:

– это платное приложение и объем бесплатно озвучиваемого текста ограничен;

– нельзя сохранить результат на сервере приложения.

Для тех, кто готов платить за озвучивание текста, «Zvukogram» несомненно является лучшим из существующих приложений для озвучивания текста. Этот сервис предоставляет чрезвычайно широкий функционал для профессионального озвучивания текста. А возможности добавления звуковых эффектов и использования нескольких голосов при озвучке, выгодно выделяют его среди аналогов.

1.3 Цели и задачи дипломного проекта. Формирование требований к приложению

1.3.1 Назначение разработки

Подводя итоги всего сказанного ранее, можно определить основные пользовательские требования и сформировать техническое задание для приложения, разрабатываемого в данном проекте.

Данное приложение предназначено для хранения и распространения аудиокниг в сети интернет, а также их синтеза на основе текста книги.

Основными целями создания данного приложения являются:

1) предоставление возможности удалённого хранения аудиокниг;

2) распространение аудиокниг в сети интернет;

3) предоставление возможности синтеза аудиокниг по текстовому файлу;

4) повышение доступности литературных произведений для людей с ограниченными возможностями.

К особенностям разрабатываемого программного средства, в сравнении с уже существующими аналогами можно отнести следующее:

– совмещение функционала приложения для хранения и распространения аудиокниг и приложения для синтеза речи;

– возможность распространить собственную версию аудиокниги среди пользователей сети Internet;

– возможность хранить аудиокниги в профиле пользователя.

1.3.2 Состав выполняемых функций:

1) Синтез аудиокниги. При синтезе аудиокниги должна присутствовать возможность настройки голоса и темпа речи. Также, должна иметься возможность скачать синтезированную аудиокнигу на устройство пользователя.

2) Регистрация. При прохождении регистрации требуется удостовериться в корректности и уникальности введенного пользователем никнейма. В случае, если пользователь с таким никнеймом уже существует, должно отобразится сообщение об ошибке. После окончания регистрации пользователь должен быть перенаправлен на авторизацию.

3) Авторизация. При прохождении авторизации требуется удостовериться в корректности введённого пользователем никнейма, а также в существовании пользователя с таким никнеймом и введённым паролем. После окончания авторизации должен быть отображен список распространяемых аудиокниг.

4) Добавление аудиокниги. При добавлении аудиокниги требуется проверить корректность данных введенных пользователем. В случае, если имеются некорректные данные, должно отобразится сообщение об ошибке. В зависимости от статуса пользователя, добавленная аудиокнига должна отобразится в списке аудиокниг, хранимых пользователем или списке распространяемых аудиокниг. Форма для добавлении аудиокниги должна содержать следующие поля:

– название книги;
– авторы книги;
– исполнители озвучки;
– жанры;
– год издания;
– краткое описание;
– поле для выбора файла обложки;
– поле выбора файла аудиокниги.

5) Отображение списка аудиокниг, хранимых пользователем. В списке аудиокниг, хранимых пользователем, должна отображаться краткая информация о каждой хранимой книге. При отображении должна присутствовать возможность сортировки и фильтрации списка аудиокниг. Кроме того, должна присутствовать возможность просмотреть и отредактировать подробную информацию об аудиокниге, а также возможность удалить аудиокнигу. И в довершение всего, должна присутствовать возможность отправить запрос на внесение аудиокниги в список распространяемых. Краткая информация об аудиокниге включает в себя:

– название книги;
– наименования авторов книги;
– наименования исполнителей озвучки;
– изображение обложки;
– жанры аудиокниги;
– дату и время добавления аудиокниги;
– рейтинг аудиокниги;
– краткое описание аудиокниги.

6) Отправка запроса на внесение аудиокниги в список распространяемых. Запрос должен содержать информацию о книге, а также никнейм пользователя и дату отправки запроса.

7) Отображение списка распространяемых аудиокниг. Список распространяемых аудиокниг должен быть доступен для любого пользователя сети Internet. В списке должна отображаться краткая информация о каждой распространяемой аудиокниге. Также, любому пользователю, должна быть доступна возможность просмотреть подробную информацию о любой аудиокниге из списка. Кроме того, для пользователя с правами администратора, должны присутствовать возможность отредактировать информацию об аудиокниге, и возможность удалить аудиокнигу.

8) Сортировка списка аудиокниг. Сортировка может осуществляться по рейтингу, названию, году издания или дате добавления аудиокниги. По окончании сортировки, порядок элементов списка должен быть изменён согласно выбранному критерию.

9) Фильтрация списка аудиокниг. По окончании фильтрации должны отображается только те, элементы списка, которые соответствуют выбранному критерию. Критериями для фильтрации могут выступать:

– название аудиокниги;
– жанр;
– наименование автора книги;
– наименование исполнителя озвучки;
– год издания.

10) Отображение подробной информации об аудиокниге. При просмотре подробной информации об аудиокниге должна присутствовать возможность скачать аудиокнигу, а также возможность её оценить. Кроме того,

должна присутствовать возможность просмотра комментариев к аудиокниге. Вдобавок, для пользователя с правами администратора, должны присутствовать возможность отредактировать информацию об аудиокниге, и возможность удалить аудиокнигу. Подробная информация об аудиокниге должна содержать:

– название книги;
– изображение обложки;
– наименования авторов книги;
– наименования исполнителей озвучки;
– жанры аудиокниги;
– год издания;
– краткое описание;
– рейтинг аудиокниги;
– дату и время добавления;
– никнейм пользователя, добавившего аудиокнигу.

11) Редактирование информации об аудиокниге. При редактировании информации об аудиокниге все поля и значения должны быть корректно загружены и отображены. До окончания редактирования должна иметься возможность отменить внесение изменений. После окончания редактирования новая версия информации об аудиокниге должна быть сохранена и отображена.

12) Удаление аудиокниги. Перед удалением пользователь должен подтвердить необходимость выполнения данной функции. По окончании удаления должен быть отображен обновлённый список аудиокниг.

13) Редактирование профиля пользователя. При редактировании профиля пользователя все поля и значения должны быть корректно загружены и отображены. Для окончания редактирования требуется получить подтверждение пользователя. После окончания редактирования новая версия профиля должна быть сохранена и отображена.

14) Скачивание аудиокниги. Перед началом скачивания должен быть отображен формат и размер скачиваемого файла.

15) Оценивание аудиокниги. По окончании оценивания, рейтинг аудиокниги должен быть пересчитан и сохранен в базе данных. После чего отображаемое значение рейтинга должно быть обновлено.

16) Просмотр профиля. В профиле пользователя должны отображаться никнейм и email. Для владельца профиля должна присутствовать возможность отредактировать профиль.

17) Выход из профиля. В приложении должна имеется возможность выйти из профиля в любой момент времени. После выхода из профиля, должен быть отображен список распространяемых книг.

18) Отображение списка запросов на внесение аудиокниги в список распространяемых. Элементы списка запросов должны содержать наименование

аудиокниги, никнейм отправителя и время отправления запроса. Должны присутствовать возможности подтвердить внесение и отказать во внесение аудиокниги в список распространяемых.

19) Подтверждение внесения аудиокниги в список распространяемых. После подтверждения внесения аудиокниги в список распространяемых, запрос на осуществление внесения должен быть удален. Внесенная аудиокнига должна отобразится в списке распространяемых.

20) Отказ во внесение аудиокниги в список распространяемых. При отказе во внесения аудиокниги в список распространяемых, запрос на осуществление внесения должен быть удален.

21) Отображение списка комментариев к аудиокниге. В списке комментариев об аудиокниге должна содержаться информация о каждом из них. Информация о комментарии содержит:

– никнейм пользователя, который написал комментарий;
– дату и время написания комментария;
– текст комментария.

22) Комментирование аудиокниги. При комментировании требуется проверить корректность введённых данных. В случае, если данные введены некорректно, комментирование считается неуспешным и должно отобразиться сообщение об ошибке. После успешного комментирования должен быть отображён обновлённый список комментариев о книге.

1.3.3 Требования к входным данным

Входные данные для приложения должны быть представлены в виде вводимого пользователем с клавиатуры текста, текстовых файлов соответствующего формата и опций, предоставляемых пользовательским интерфейсом приложения. В приложении должны быть реализованы проверки входных данных на корректность, и, в случае их некорректности, пользователь должен получать соответствующее уведомление с просьбой ввести данные необходимого формата.

1.3.4 Требования к выходным данным

Выходные данные должны быть представлены в виде аудиофайлов соответствующего формата, файловых архивов, содержащих аудиофайлы, а также посредством отображения информации при помощи различных элементов пользовательского интерфейса.

1.3.5 Требования к надежности

Для обеспечения надежности приложения требуется обеспечить бесперебойное питания технического средства, и своевременные проверки оборудования на наличие вирусных программ.

Время восстановления после отказа, вызванного сбоем электропитания технических средств, или не фатальным сбоем операционной системы, не

должно превышать 60-ти минут при условии соблюдения условий эксплуатации технических и программных средств. Время восстановления после отказа, вызванного неисправностью технических средств, или фатальным сбоем операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

Веб-приложение не должно непредвиденно прерывать свою работу. Отказы приложения вследствие некорректных действий пользователя при взаимодействии с приложение через веб-интерфейс недопустимы.

1.3.6 Технические требования

Архитектура всей системы должна отвечать следующим требованиям:

1) централизованная база данных;

2) организация доступа к компонентам системы через внешний канал связи (Internet);

3) разделение бизнес логики, обработки и представления данных;

4) безопасность;

5) надёжность.

Для обеспечения работы системы требуются технические средства для размещения базы данных и серверной части системы. Должна быть обеспечена круглосуточная работа приложения.

Требования к техническому обеспечению серверной части системы:

1) процессор Intel Core i5 с тактовой частотой 2 ГГц или более мощный;

2) оперативная память в объеме 4Гбайт или более;

3) свободное место на жестком диске в объеме не менее 20 ГБ;

4) постоянное подключение к сети Internet.

Требования к программному обеспечению серверной части:

1) ОС Windows Server 2019 версии 10.0 или выше;

2) СУБД MySQL версии 8.0 или выше.

Требования к техническому и программному обеспечению устройства клиента:

1) стабильное подключение к сети Internet;

2) браузер с поддержкой HTML5 и JavaScript.

2.1.2 Разработка инфологической модели базы данных

Исходя из необходимости использования в проектируемом приложении базы данных, разработаем ее инфологическую модель. Для создания данной модели возьмем за основу предметную область проекта.

Предметная область разрабатываемого программного средства включает в себя следующие сущности и их атрибуты:

1) пользователь:
   - уникальный идентификатор;
   - никнейм;
   - хешированный пароль;
   - адрес электронной почты;
   - идентификатор роли пользователя;
2) роль пользователя:
   - уникальный идентификатор;
   - наименование роли;
3) аудиокнига:
   - уникальный идентификатор;
   - идентификатор пользователя, добавившего аудиокнигу;
   - является ли книга распространяемой;
   - название книги;
   - путь к файлу содержащему изображение обложки;
   - год издания;
   - краткое описание;
   - дата добавления;
   - идентификатор файла аудиокниги;
   - рейтинг аудиокниги;
4) аудиокнига, хранимая пользователем:
   - идентификатор пользователя;
   - идентификатор аудиокниги;
5) оценка:
   - идентификатор пользователя;
   - идентификатор книги;
   - значение;
6) жанр:
   - уникальный идентификатор;
   - наименование жанра;
7) создатель аудиокниги:
   - уникальный идентификатор;
   - наименование создателя;
8) жанр аудиокниги:
   - идентификатор жанра;
   - идентификатор аудиокниги;

9) создатель:
- уникальный идентификатор;
- наименование создателя;
- является ли автором;

10) файл аудиокниги:
- уникальный идентификатор;
- расширение файла;
- размер файла;
- путь к файлу содержащему аудиокнигу;

11) комментарий к аудиокниге:
- уникальный идентификатор;
- идентификатор пользователя;
- идентификатор аудиокниги;
- дата и время добавления;
- текст комментария;

12) запрос на внесение аудиокниги в список распространяемых:
- уникальный идентификатор;
- идентификатор пользователя;
- идентификатор аудиокниги;
- дата и время отправления запроса.

2.2 Спецификация функциональных требований

С учетом требований, определенных в подразделе 1.3, представим детализацию функций проектируемого ПС.

Для детализации функций рассмотрим основные требования, предъявляемые к каждой функции программного средства как с точки зрения внутренней организации системы, так и с точки зрения взаимодействия системы с пользователем.

2.2.1 Синтез аудиокниги

Функция синтеза аудиокниги должна быть реализована с учетом следующих требований:

1) процесс синтеза аудиокниги может быть инициирован пользователем системы со статусом «User» или «Admin»;

2) должна присутствовать возможность настройки параметров работы синтезатора речи, а именно скорости произношения и высоты интонации;

3) должна присутствовать возможность выбора голоса, используемого при синтезе, из списка доступных;

4) должна присутствовать возможность синтезировать и прослушать пробный текст с настройками синтезатора, установленными пользователем;

5) для синтеза аудиокниги пользователь должен предоставить текстовый файл в формате txt, размером до 900 мегабайт;

6) результатом синтеза является аудиофайл в формате mp3, или архив в формате zip, содержащий несколько mp3 файлов;

7) должна присутствовать возможность скачать синтезированную аудиокнигу на устройство пользователя.

2.2.2 Регистрация

Функция регистрации должна быть реализована с учетом следующих требований:

1) процесс регистрации инициируется пользователем системы со статусом «Guest»;

2) для регистрации пользователь должен предоставить уникальный никнейм, а также дважды ввести пароль;

3) корректность введенных данных должна быть проверена при помощи встроенных инструментов разработки;

4) корректным считается никнейм состоящий из латинских символов и цифр, а также символа «_», длинной не более 35 символов;

5) корректным паролем считается последовательность, состоящая из латинских символов и цифр, а также символа «_», длиной от 3 до 20 символов;

6) в случае некорректности введенных данных пользователь должен увидеть сообщение об этом с предложением попробовать еще раз;

7) необходимо удостовериться в уникальность введенного никниейма;

8) в случае если никнейм не является уникальным пользователь должен увидеть сообщение об этом с предложением изменить никнейм;

9) по окончании регистрации пользователь должен быть перенаправлен на авторизацию.

2.2.3 Авторизация

Функция регистрации должна быть реализована с учетом следующих требований:

1) процесс авторизации инициируется пользователем системы со статусом «Guest»;

2) для прохождения авторизации пользователь должен ввести свой никнейм и пароль;

3) корректность введенных данных должна быть проверена при помощи встроенных инструментов разработки;

4) корректным считается никнейм состоящий из латинских символов и цифр, а также символа «_», длинной не более 35 символов;

5) корректным паролем считается последовательность, состоящая из латинских символов и цифр, а также символа «_», длиной от 3 до 20 символов.

6) в случае некорректности введенных данных пользователь должен увидеть сообщение об этом с предложением попробовать еще раз;

7) необходимо удостовериться в существовании пользователя, с никнейм и паролем, соответствующим введенным;

8) в случае, если пользователь с никнеймом и паролем, соответствующим введенным, не существует, должно отобразиться сообщение некорректности введенных данных;

9) по окончании авторизации пользователя должен быть отображен список распространяемых аудиокниг;

10) по окончании авторизации статус пользователя должен быть изменен на «User» или «Admin», в зависимости от ассоциированной с его профилем роли.

2.2.4 Добавление аудиокниги

Функция добавления аудиокниги должна быть реализована с учетом следующих требований:

1) процесс добавления аудиокниги может быть инициирован пользователем системы со статусом «User» или «Admin»;

2) для добавления аудиокниги пользователь должен предоставить следующую информацию:

- название аудиокниги;
- наименования авторов аудиокниги;
- наименования исполнителей аудиокниги;
- жанры книги;
- год издания;
- краткое описание;

3) должна присутствовать возможность загрузить обложку аудиокниги, обложкой является файл изображения в формате png или jpg;

4) для добавления аудиокниги пользователь должен предоставить файл содержащий аудиокнигу, это может быть mp3 файл или архив в формате zip;

5) название книги представляет собой последовательность длинной от 3 до 70 символов состоящую из букв кириллического и латинского алфавитов, а также знаков препинания и пробельных символов;

6) должна присутствовать возможность ассоциировать с книгу с одним или несколькими авторами;

7) наименование автора книги, это последовательность длинной от 3 до 40 символов состоящая из букв кириллического алфавита, знаков препинания и пробельных символов;

8) если в базе данных приложения отсутствует автор с наименованием, соответствующим введённому пользователем, тогда в базу данных должен быть добавлен новый автор с соответствующим наименованием;

9) должна присутствовать возможность ассоциировать с книгу с одним или несколькими исполнителями;

10) наименование исполнителя аудиокниги, это последовательность длинной от 3 до 40 символов состоящая из букв кириллического алфавита, знаков препинания и пробельных символов;

11) если в базе данных приложения отсутствует исполнитель с наименованием, соответствующим введённому пользователем, тогда в базу данных должен быть добавлен новый исполнитель с соответствующим наименованием;

12) должна присутствовать возможность ассоциировать книгу с одним или несколькими жанрами, выбираемыми из списка доступных на сайте;

13) год издания представляет собой число в промежутке 1800 до 2021;

14) краткое описание представляет собой последовательность длинной от 0 до 2000 символов состоящую из букв кириллического алфавита, знаков препинания и пробельных символов;

15) данные, введенные пользователем, должны проверяться на корректность, если имеются некорректные данные, должно отобразиться сообщение об ошибке;

16) аудиокнига, добавленная пользователем со статусом «User» должна отобразиться в списке аудиокниг, хранимых пользователем;

17) аудиокнига, добавленная пользователем со статусом «Admin» должна отобразиться в списке распространяемых аудиокниг.

2.2.5 Отображение списка аудиокниг, хранимых пользователем

Функция отображения списка аудиокниг, хранимых пользователем должна быть реализована с учетом следующих требований:

1) доступом к работе со списком хранимых книг должны обладать только пользователи со статусом «User»;

2) в списке аудиокниг, хранимых пользователем, должна отображаться краткая информация о каждой хранимой аудиокниге;

3) краткая информация о хранимой аудиокниге включает в себя:

- название книги;
- от 1 до 5 наименований авторов книги;
- от 1 до 5 наименований исполнителей озвучки;
- изображение обложки;
- от 1 до 8 жанров аудиокниги;
- дату и время добавления аудиокниги;
- рейтинг аудиокниги;
- краткое описание аудиокниги;

4) рейтинг аудиокниги представляет собой десятичную дробь с округлением до сотых, и значением в промежутке от 1 до 10;

5) должна быть реализована пагинация списка аудиокниг с отображением:

- текущей страницы;
- первой страницы пагинации;
- последней страниц пагинации;

6) на одной странице пагинации должно отображаться не более 10 аудиокниг;

7) по нажатии на название аудиокниги или изображение обложки должна быть отображена подробная информация о выбранной книге;

8) для каждого элемента списка должна присутствовать возможность инициировать редактирование информации о книге;

9) для каждого элемента списка должна присутствовать возможность инициировать удаление аудиокниги;

10) для тех аудиокниг, которые не являются распространяемыми, должна присутствовать возможность отправить запрос на внесение книги в список распространяемых;

11) должна присутствовать возможность фильтрации списка;

12) должна присутствовать возможность сортировки списка.

2.2.6 Отправка запроса на внесение аудиокниги в список распространяемых

Функция отправки запроса на внесение аудиокниги в список распространяемых должна быть реализована с учетом следующих требований:

1) возможность инициировать отправку запроса должна иметься только у пользователей со статусом «User»;

2) запрос должен содержать дату и время отправления, а также информацию о книге и никнейм пользователя;

3) после подтверждения отправки, запрос должен быть сохранен в базе данных.

2.2.7 Отображение списка распространяемых аудиокниг

Функция отображения списка распространяемых аудиокниг должна быть реализована с учетом следующих требований:

1) просмотр списка распространяемых аудиокниг, должен быть доступен любому пользователю независимо от статуса;

2) в списке распространяемых аудиокниг должна отображаться краткая информация о каждой распространяемой аудиокниге;

3) краткая информация о распространяемой аудиокниге включает в себя:

- название книги;
- от 1 до 5 наименований авторов книги;
- от 1 до 5 наименований исполнителей озвучки;
- изображение обложки;
- от 1 до 8 жанров аудиокниги;
- длительность аудиокниги;
- рейтинг аудиокниги;
- краткое описание аудиокниги;
- никнейм пользователя, добавившего книгу;

4) рейтинг аудиокниги представляет собой десятичную дробь с округлением до сотых, и значением в промежутке от 1 до 10;

5) должна быть реализована пагинация списка с отображением:

– текущей страницы;

– первой страницы пагинации;

– последней страниц пагинации;

6) на одной странице пагинации должно отображаться не более 10 книг;

7) по нажатии на название аудиокниги или изображение обложки должна быть отображена подробная информация о выбранной книге;

8) для каждого элемента списка пользователю со статусом «Admin» должна быть доступна возможность инициировать редактирование информации о книге;

9) для каждого элемента списка пользователю со статусом «Admin» должна быть доступна возможность инициировать удаление аудиокниги из списка распространяемых.

10) должна присутствовать возможность фильтрации списка;

11) должна присутствовать возможность сортировки списка.

2.2.8 Сортировка списка аудиокниг

Функция сортировки списка аудиокниг должна быть реализована с учетом следующих требований:

1) сортировка должна выполняться на основе одного из следующих критериев:

– название книги;

– рейтинг;

– год издания;

– дата добавления;

2) должна присутствовать возможность сортировки как по возрастанию, так и по убыванию выбранного критерия;

3) по окончании сортировки должен быть отображён список, порядок элементов которого изменен согласно параметрам сортировки.

2.2.9 Фильтрация списка аудиокниг

Функция фильтрации списка аудиокниг должна быть реализована с учетом следующих требований:

1) фильтрация должна выполняться на основе введённых или выбранных пользователем критериев;

2) критериями для фильтрации могут выступать:

– название книги;

– жанр;

– год издания;

– наименование автора;

– наименование исполнителя;

3) фильтрация по введённому пользователем названию книги должна выполняться на основе частичного или полного соответствия названию

аудиокниги, представленной в списке, без учёта регистра;

4) фильтрация по жанру должна выполняться на основе наличия у книги жанра, который пользователь выбрал из списка доступных в приложении;

5) фильтрация по наименованию автора должна выполняться на основе частичного или полного соответствия одному из наименований авторов, ассоциированных с аудиокнигой, представленной в списке;

6) фильтрация по наименованию исполнителя должна выполняться на основе частичного или полного соответствия одному из наименований исполнителей, ассоциированных с аудиокнигой, представленной в списке;

7) фильтрация по году должна выполняться на основе совпадения года издания книги с годом, введенным пользователем;

8) по окончании сортировки должен быть отображён список, все элементы которого соответствуют выбранным критериям.

2.2.10 Отображение подробной информации об аудиокниге

Данная функция должна быть реализована с учётом следующих требований:

1) подробная информация об аудиокниге включает в себя:

- название книги;
- наименования авторов книги;
- наименования исполнителей озвучки;
- изображение обложки;
- все жанры аудиокниги;
- год издания;
- краткое описание;
- рейтинг аудиокниги;
- длительность аудиокниги;
- дату и время добавления;
- никнейм пользователя, добавившего аудиокнигу;

2) дата и время добавления должны быть представлены в формате «dd/mm/yyyy HH:MM:SS», где «dd» - двухзначный день месяца, «mm» - двухзначный номер месяца, «yyyy» - четырёхзначный номер года, «HH» - количество часов, «MM» - количество минут, «SS» - количество секунд;

3) для пользователя со статусом «User» должна присутствовать возможность оценить книгу;

4) должна присутствовать возможность просмотреть список комментариев о книге;

5) должна присутствовать возможность скачать аудиокнигу.

2.2.11 Редактирование информации об аудиокниге

Данная функция должна быть реализована с учётом следующих требований:

1) процесс редактирования информации об аудиокниге может быть инициирован либо пользователем, добавившим аудиокнигу, либо пользователем со статусом «Admin»;

2) при редактировании информации об аудиокниге все поля и значения должны быть корректно загружены и отображены;

3) информация об аудиокниге доступная для редактирования:

- название книги;
- наименования авторов книги;
- наименования исполнителей аудиокниги;
- изображение обложки;
- жанры аудиокниги;
- год издания;
- краткое описание;

4) должна присутствовать возможность загрузить новую обложку аудиокниги, обложкой является файл изображения в формате png или jpg;

5) название книги представляет собой последовательность длинной от 3 до 70 символов состоящую из букв кириллического и латинского алфавитов, а также знаков препинания и пробельных символов;

6) должна присутствовать возможность ассоциировать с книгу с одним или несколькими новыми авторами, а также удалить уже существующие ассоциации;

7) наименование автора, это последовательность длинной от 3 до 40 символов состоящая из букв кириллического алфавита, знаков препинания и пробельных символов;

8) если в базе данных приложения отсутствует автор с наименованием, соответствующим введённому пользователем, тогда в базу данных должен быть добавлен новый автор с соответствующим наименованием;

9) должна присутствовать возможность ассоциировать с книгу с одним или несколькими новыми исполнителями, а также удалить уже существующие ассоциации;

10) наименование исполнителя аудиокниги, это последовательность длинной от 3 до 40 символов состоящая из букв кириллического алфавита, знаков препинания и пробельных символов;

11) если в базе данных приложения отсутствует исполнитель с наименованием, соответствующим введённому пользователем, тогда в базу данных должен быть добавлен новый исполнитель с соответствующим наименованием;

12) должна присутствовать возможность ассоциировать книгу с одним или несколькими жанрами, выбираемыми из списка доступных на сайте, а также удалить уже существующие ассоциации;

13) год издания представляет собой число в промежутке 1800 до 2100;

14) краткое описание представляет собой последовательность длинной от 0 до 2000 символов состоящую из букв кириллического алфавита, знаков

препинания и пробельных символов;

15) данные, введенные пользователем, должны проверяться на корректность, если имеются некорректные данные, должно отобразиться сообщение об ошибке;

16) до окончания редактирования должна иметься возможность отменить внесение изменений;

17) по окончании редактирования новая версия информации об аудиокниге должна быть сохранена в базу данных;

18) по окончании редактирования должна быть отображена обновленная версия подробной информации об аудиокниге.

2.2.12 Удаление аудиокниги

Данная функция должна быть реализована с учётом следующих требований:

1) процесс удаления аудиокниги может быть инициирован либо пользователем, добавившим аудиокнигу, либо пользователем со статусом «Admin»;

2) по окончании удаления должен быть отображен обновлённый список аудиокниг.

2.2.13 Редактирование профиля пользователя

Данная функция должна быть реализована с учётом следующих требований:

1) процесс редактирования профиля может быть инициирован только тем пользователем, которому принадлежит профиль;

2) при редактировании профиля пользователя все поля и значения должны быть корректно загружены и отображены;

3) информация о пользователе доступная для редактирования:

– никнейм;

– email;

– пароль;

4) данные, введенные пользователем, должны проверяться на корректность.

5) корректным считается никнейм состоящий из латинских символов и цифр, а также символа «_», длинной не более 35 символов;

6) корректным паролем считается последовательность, состоящая из латинских символов и цифр, а также символа «_», длиной от 3 до 20 символов;

7) корректным значением поля email является последовательность длинной до 64 символов, состоящая из букв латинского алфавита и цифр с добавлением в конец символа «@» и доменного имени (пример: mail.ru);

8) поле email является необязательным для заполнения;

9) в случае некорректности введенных данных пользователь должен увидеть сообщение об этом;

10) необходимо удостовериться в уникальность введенного никниейма;

11) в случае если никнейм не является уникальным пользователь должен увидеть сообщение об этом с предложением изменить никнейм;

12) до окончания редактирования должна иметься возможность отменить внесение изменений;

13) по окончании редактирования новая версия профиля должна быть сохранена в базу данных;

14) по окончании редактирования должна быть отображена обновленная версия профиля пользователя.

2.2.14 Скачивание аудиокниги

Данная функция должна быть реализована с учётом следующих требований:

1) перед началом скачивания должен быть отображен формат и размер скачиваемого файла;

2) файлом аудиокниги является аудиофайл в формате mp3, или архив формата zip, содержащий несколько файлов;

3) максимальный размер файла, доступного для скачивания, составляет 900 мегабайт.

2.2.15 Оценивание аудиокниги

Данная функция должна быть реализована с учётом следующих требований:

1) оценивать книги может только пользователь со статусом «User»;

2) по окончании оценивания, рейтинг аудиокниги должен быть пересчитан и сохранен в базе данных, после чего отображаемое значение рейтинга должно быть обновлено.

2.2.16 Просмотр профиля

Данная функция должна быть реализована с учётом следующих требований:

1) в профиле пользователя должны отображаться никнейм и email;

2) должна присутствовать возможность отредактировать профиль;

2.2.17 Выход из профиля

Данная функция должна быть реализована с учётом следующих требований:

1) выход из профиля может быть инициирован пользователями со статусом «User» или «Admin»;

2) возможность покинуть профиль должна присутствовать в любой момент времени, независимо от функции выполняемой приложением;

3) после выхода из профиля, статус пользователя должен смениться на «Guest»;

4) после выхода из профиля, должен быть отображен список распространяемых книг.

2.2.18 Отображение списка запросов на внесение аудиокниги в список распространяемых

Данная функция должна быть реализована с учётом следующих требований:

1) доступ к работе со списком запросов имеют только те пользователи, которые обладают статусом «Admin»;

2) элементы списка запросов должны содержать название аудиокниги, никнейм отправителя и время отправления запроса;

3) должна быть реализована пагинация списка с отображением:

– текущей страницы;

– первой страницы пагинации;

– последней страниц пагинации;

4) на одной странице пагинации должно отображаться не более 10 запросов;

5) при нажатии на название книги должна быть отображена подробная информация об указанной аудиокниге;

6) должна присутствовать возможность отказать во внесении аудиокниги в список распространяемых;

7) должна присутствовать возможность подтвердить внесение аудиокниги в список распространяемых.

2.2.19 Подтверждение внесения аудиокниги в список распространяемых

Данная функция должна быть реализована с учётом следующих требований:

1) доступ к данной функции имеют только те пользователи, которые обладают статусом «Admin»;

2) после подтверждения внесения аудиокниги в список распространяемых, статус аудиокниги в базе данных должен быть изменен;

3) после подтверждения внесения аудиокниги в список распространяемых, запрос на осуществление внесения должен быть удален, а отображение списка запросов обновлено.

2.2.20 Отказ во внесение аудиокниги в список распространяемых

Данная функция должна быть реализована с учётом следующих требований:

1) доступ к работе к данной функции имеют только те пользователи, которые обладают статусом «Admin»;

2) при отказе во внесении аудиокниги в список распространяемых, запрос на осуществление внесения должен быть удален, а отображение списка запросов обновлено.

2.2.21 Отображение списка комментариев к аудиокниге

Данная функция должна быть реализована с учётом следующих требований:

1) в списке комментариев должна отображаться информация о каждом из них;

2) информация о комментарии содержит:

– никнейм пользователя, который добавил комментарий;

– дату написания комментария;

– текст комментария;

3) список комментариев должен быть отсортирован по возрастанию даты написания комментария;

2.2.22 Комментирование аудиокниги

Данная функция должна быть реализована с учётом следующих требований:

1) процесс комментирования аудиокниги может быть инициирован только пользователем со статусом «User»;

2) для комментирования обязательно требуется ввести текст комментария, который является последовательностью длинной от 1 до 500 символов, состоящий из букв латинского или кириллического алфавитов, знаков препинания и пробельных символов;

3) данные, введенные пользователем, должны проверяться на корректность, и если имеются некорректные данные, то должно отобразиться сообщение об ошибке;

4) после окончания комментирования отображаемый список комментариев должен быть обновлён.

3 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ

3.1 Разработка архитектуры приложения

Для написания любого проекта очень важно чтобы приложение не только правильно и быстро работало, но и было правильно организованно. Сложность, как правило, растет гораздо быстрее размеров программы. И если не позаботиться об этом заранее, то вскоре наступит момент, когда контролировать приложение станет невозможно.

Правильно подобранная архитектура экономит очень много сил, времени и ресурсов. Хорошая архитектура – это, прежде всего, выгодная архитектура, делающая процесс разработки и сопровождения программы более простым и эффективным. Приложение с правильно подобранной архитектурой легче расширять и изменять, а также тестировать, отлаживать и описывать.

Поэтому, как только были сформулированы основные функциональные требования проектируемого приложения, требуется разработать архитектуру приложения.

Качественная архитектура должна обладать следующими свойствами:

1) функциональность. Данное свойство представляет собой возможность реализации функциональных требований, указанных ранее, на основе данной архитектуры;

2) гибкость системы. Это свойство означает, что качественная архитектура должна обеспечить возможность быстрого и качественного внесения изменений в приложения, а также минимизировать количество ошибок, вызываемых изменениями приложения;

3) возможность независимого изменения. Это свойство заключается в том, что изменения, вносимые в работу одного из компонентов приложения, не должны затрагивать деятельность иных компонентов

4) удобство построения. Данное свойство говорит о том, что архитектура должна обеспечить возможность построения приложения таким образом, чтобы набор компонентов приложения мог реализовываться и тестироваться независимо друг от друга;

5) расширяемость. Возможность добавлять в систему новые сущности и функции, не нарушая ее основной структуры;

6) сопротивление энтропии – поддержание упорядоченности системы за счёт принятия, ограничения и изоляции последствий изменений;

7) модульность. Данное свойство определяет возможность приложения делиться на рабочие задания (модули), и особенно модули, которые могут разрабатываться независимо друг от друга, при этом легко и точно дополняя возможности друг друга;

8) безопасность – качественная архитектура предоставляет приложению возможность управления ограничением доступа к своим данным;

9) многоразовость компонентов – возможность повторного использования модулей системы.

Модель, в контексте паттерна MVC, представляет данные и методы работы с ними. В веб-приложения, модель, как правило, отвечает за работу с базами данных и файловой системой. По сути, модель просто предоставляет данные предметной области представлению и реагирует на команды контроллера, изменяя свое состояние.

За отображение данных, которые отдаёт модель, несёт ответственность представление. Зачастую представление описывается как шаблон, заполняемый данными. Оно позволяет менять внешний вид отображаемой информации, но не оказывает влияния на саму информацию. В веб-приложениях представление зачастую реализуется в виде HTML-страницы, но иногда может быть представлено данными в формате JSON или XML.

Модель и представление связываются контроллером. Контроллер получает запрос от клиента, анализирует его параметры и для выполнения операций над данными запроса обращается к модели. От модели поступают уже скомпонованные объекты. Затем они отправляются в представление, которое передаёт сформированную страницу контроллеру, а он, в свою очередь, отправляет её клиенту [12].

Дополнительными плюсам данной архитектуры также являются:

– стандартизация кодирования;

– лёгкость обнаружения и исправления ошибок;

– быстрое вхождение в проект новых разработчиков;

– лёгкое изменение способа хранения сущностей.

Использование шаблона MVC значительно облегчает разработку приложения с клиент-серверной архитектурой, поскольку сразу создает хорошо обозначенное разграничение между основными субъектами данного подхода: клиентом и сервером. К тому же MVC позволяет качественно и быстро отладить и протестировать приложение.

Архитектуру MVC на языке Java поддерживает такой фреймворк как Spring MVC. Он представляет собой веб-службу, которая может взаимодействовать с различными приложениями. При этом приложение может быть, как веб-приложением на базе Spring, так и мобильным или обычным десктопным приложением.

Все запросы, которые обрабатываются Spring MVC приложением, проходят путь, показанный на рисунке 3.2:

1) клиентское приложение создаёт запрос к серверу;

2) HTTP-сервер загружает конфигурацию, создаёт экземпляр приложения;

3) с помощью объекта для управления запросами определяется контроллер;

4) для обработки запроса приложение создаёт экземпляр контроллера;

5) контроллер находит нужное действие и выполняет фильтры для действия;

6) действие не выполняется при неудачном выполнении любого фильтра;

7) действие выполняется при успешном выполнении всех фильтров;

8) действие загружает модель данных;

9) действие возвращает данные контроллеру;

10) контроллер сериализует данные и помещает их в тело ответа;

11) сервер формирует и отсылает ответ на запрос клиентского приложения.

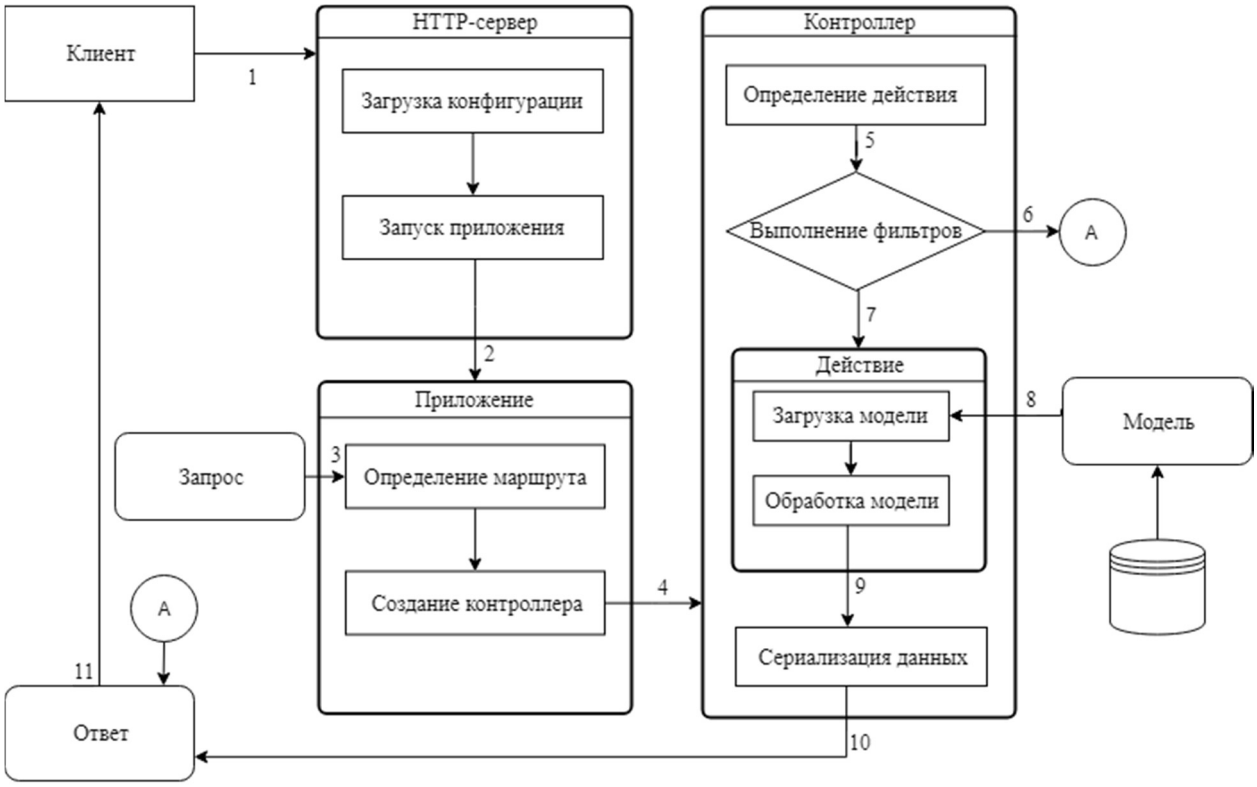

Рисунок 3.2 – Диаграмма обработки запроса серверной частью приложения

Таким образом для разработки серверной части данного приложения прекрасно подходит паттерн MVC совмещённой с клиент-серверной архитектурой. Объединение MVC паттерна с клиент-серверной архитектурой, позволяет значительно снизить нагрузку на сервер приложения, перенеся её часть на клиентские аппараты. Учитывая, что сейчас мобильные гаджеты и персональные компьютеры становятся всё мощнее перенос части нагрузки на них не взывает ухудшения работы разрабатываемого приложения.

3.2 Разработка даталогической и физической моделей базы данных

На даталогическом уровне модель предметной области представляется в привязке к конкретной модели данных и описывает способ организации данных безотносительно их физического размещения в конкретной СУБД. В приложении, описываемом данным дипломным проектом, будет использована реляционная модель данных.

Реляционная модель данных представляет собой интуитивно понятный, наглядный табличный способ представления данных, где каждая строка, со-

держащаяся в таблице, представляет собой запись с уникальным идентификатором, который именуется ключом. Столбцы таблицы имеют атрибуты данных, а каждая запись обычно содержит значение для каждого атрибута, что дает возможность легко устанавливать взаимосвязь между элементами данных [13].

В реляционной модели данных логическая структура (таблицы, представления и индексы) отличается от физической структуры хранения. Благодаря этому создаётся возможность управления физической системой хранения, не меняя данных, которые содержатся в логической структуре. Например, изменение имени файла базы данных не повлияет на хранящиеся в нем таблицы.

Модель базы данных даталогического уровня, представлена в таблицах 3.1-3.12.

Таблица 3.1 – Даталогическая структура сущности «Audiobook»

Атрибут	Значение атрибута	Тип атрибута
Id	Уникальный идентификатор	Числовой
User_Id	Идентификатор пользователя, добавившего аудиокнигу	Числовой
Distributed	Является ли книга распространяемой	Логический
Title	Название книги	Текстовый
Picture_Path	Путь к файлу содержащему изображение обложки	Текстовый
Publication_Year	Год издания	Числовой
Description	Краткое описание	Текстовый
Add_Date	Дата добавления	Дата/Время
Audiobook_FileId	Идентификатор файла аудиокниги	Числовой
Rating	Рейтинг аудиокниги	Числовой

Сущность «Audiobook», представленная в таблице 3.1, предназначена для хранения информации об аудиокниге. Первичным ключом данной сущности является поле Id, хранящее уникальный идентификатор аудиокниги. Сущность «Audiobook» связана с сущностью «User» через поле «User_Id», хранящее уникальный идентификатор пользователя, добавившего аудиокнигу в приложение. Также данная сущность связана с сущность «Audiobook_File» через поле «Audiobook_FileId», хранящее уникальный идентификатор файла аудиокниги.

Таблица 3.2 – Даталогическая структура сущности «Audiobook_File»

Атрибут	Значение атрибута	Тип атрибута
Id	Уникальный идентификатор	Числовой
Size	Размер файла	Текстовый
Extension	Расширение файла	Текстовый
File_Path	Путь к файлу содержащему аудиокнигу	Текстовый

Сущность «Audiobook_File», представленная в таблице 3.2, предназначена для хранения информации о файле хранящем аудиокнигу. Первичным ключом данной сущности является поле Id, хранящее уникальный идентификатор файла.

Таблица 3.3 – Даталогическая структура сущности «Creator»

Атрибут	Значение атрибута	Тип атрибута
Id	Уникальный идентификатор	Числовой
Title	Наименование создателя	Текстовый
Author	Является ли автором книги	Логический

Сущность «Creator», представленная в таблице 3.3, предназначена для хранения информации о создателе аудиокниги. Первичным ключом данной сущности является поле Id, хранящее уникальный идентификатор создателя аудиокниги.

Таблица 3.4 – Даталогическая структура сущности «User»

Атрибут	Значение атрибута	Тип атрибута
Id	Уникальный идентификатор	Числовой
Nickname	Уникальное имя пользователя	Текстовый
Email	Электронная почта	Текстовый
Password	Хэш значение пароля	Текстовый
Role_Id	Идентификатор роли пользователя	Числовой

Сущность «User», представленная в таблице 3.4, предназначена для хранения информации о профиле пользователя. Первичным ключом данной сущности является поле Id, хранящее уникальный идентификатор пользователя. Сущность «User» связана с сущностью «Role» через поле «Role_Id», хранящее уникальный идентификатор роли.

Таблица 3.5 – Даталогическая структура сущности «Genre»

Атрибут	Значение атрибута	Тип атрибута
Id	Уникальный идентификатор	Числовой
Title	Наименование жанра	Текстовый

Сущность «Genre», представленная в таблице 3.5, предназначена для хранения информации о жанре аудиокниги. Первичным ключом данной сущности является поле Id, хранящее уникальный идентификатор жанра.

Таблица 3.6 – Даталогическая структура сущности «Role»

Атрибут	Значение атрибута	Тип атрибута
Id	Уникальный идентификатор	Числовой
Title	Наименование роли	Текстовый

Сущность «Role», представленная в таблице 3.6, предназначена для хранения информации о роли пользователя в системе. Первичным ключом данной сущности является поле Id, хранящее уникальный идентификатор роли пользователя.

Таблица 3.7 – Даталогическая структура сущности «Audiobook_Genre»

Атрибут	Значение атрибута	Тип атрибута
Genre_Id	Идентификатор жанра	Числовой
Audiobook_Id	Идентификатор аудиокниги	Числовой

Сущность «Audiobook_Genre», представленная в таблице 3.7, предназначена для хранения информации о том, к какими жанрам относится аудиокнига. Сущность «Audiobook_Genre» связывает между собой сущности «Genre» и «Audiobook». Первичным ключом данной сущности является уникальное сочетание полей «Genre_Id» и «Audiobook_Id», хранящих уникальный идентификатор жанра и уникальный идентификатор аудиокниги соответственно.

Таблица 3.8 – Даталогическая структура сущности «Audiobook_Creator»

Атрибут	Значение атрибута	Тип атрибута
Creator_Id	Идентификатор создателя аудиокниги	Числовой
Audiobook_Id	Идентификатор аудиокниги	Числовой

Сущность «Audiobook_Creator», представленная в таблице 3.8, предназначена для хранения информации о тех, кем была создана аудиокнига. Сущность «Audiobook_Creator» связывает между собой сущности «Creator» и «Audiobook». Первичным ключом данной сущности является уникальное сочетание полей «Creator_Id» и «Audiobook_Id», хранящих уникальный идентификатор создателя и уникальный идентификатор аудиокниги соответственно.

Таблица 3.9 – Даталогическая структура сущности «Query»

Атрибут	Значение атрибута	Тип атрибута
Id	Уникальный идентификатор	Числовой
User_Id	Идентификатор пользователя	Числовой
Audiobook_Id	Идентификатор аудиокниги	Числовой
Send_DateTime	Дата и время отправления запроса	Дата/Время

Сущность «Query», представленная в таблице 3.9, предназначена для хранения информации о запросе на внесение аудиокниги в список распространяемых. Первичным ключом данной сущности является поле Id, хранящее уникальный идентификатор запроса. Сущность «Query» связана с сущностью «User» через поле «User_Id», хранящее идентификатор пользователя, отправившего запрос. А также, с сущностью «Audiobook» через поле «Audiobook_Id», хранящее идентификатор аудиокниги, добавляемой в список распространяемых.

Таблица 3.10 – Даталогическая структура сущности «User_Audiobook»

Атрибут	Значение атрибута	Тип атрибута
User_Id	Идентификатор пользователя	Числовой
Audiobook_Id	Идентификатор аудиокниги	Числовой

Сущность «User_Audiobook», представленная в таблице 3.10, предназначена для хранения информации о том какие аудиокниги хранятся в профиле пользователя. Сущность «User_Audiobook» связывает между собой сущности «User» и «Audiobook». Первичным ключом данной сущности является уникальное сочетание полей «User_Id» и «Audiobook_Id», хранящих уникальный идентификатор профиля пользователя и уникальный идентификатор аудиокниги соответственно.

Таблица 3.11 – Даталогическая структура сущности «Rating»

Атрибут	Значение атрибута	Тип атрибута
User_Id	Идентификатор пользователя	Числовой
Audiobook_Id	Идентификатор книги	Числовой
Value	Значение оценки	Числовой

Сущность «Rating», представленная в таблице 3.11, предназначена для хранения информации об оценке, которую пользователь выставил аудиокниге. Первичным ключом данной сущности является уникальное сочетание полей «User_Id» и «Audiobook_Id», хранящих уникальный идентификатор создателя и уникальный идентификатор аудиокниги соответственно.

Таблица 3.12 – Даталогическая структура сущности «Comment»

Атрибут	Значение атрибута	Тип атрибута
Id	Уникальный идентификатор	Числовой
User_Id	Идентификатор пользователя	Числовой
Audiobook_Id	Идентификатор аудиокниги	Числовой
Text	Текст комментария	Текстовый
Send_DateTime	Дата и время добавления комментария	Дата/Время

Сущность «Comment», представленная в таблице 3.12, предназначена для хранения информации о комментариях к книге. Первичным ключом сущности является поле Id, хранящее уникальный идентификатор комментария. Сущность «Comment» связана с сущностью «User» через поле «User_Id», хранящее идентификатор пользователя, отправившего запрос. А также, с сущностью «Audiobook» через поле «Audiobook_Id», хранящее идентификатор аудиокниги.

Физический уровень моделирования БД описывает конкретные таблицы, связи, индексы, методы хранения, настройки производительности, безопасности. Описывать физическую модель требуется с помощью терминов и типов данных присущих конкретной СУБД.

3.3 Разработка алгоритма приложения и алгоритмов отдельных модулей

Общий алгоритм работы приложения представлен в виде схемы программы, наблюдать которую можно на рисунке 3.3.

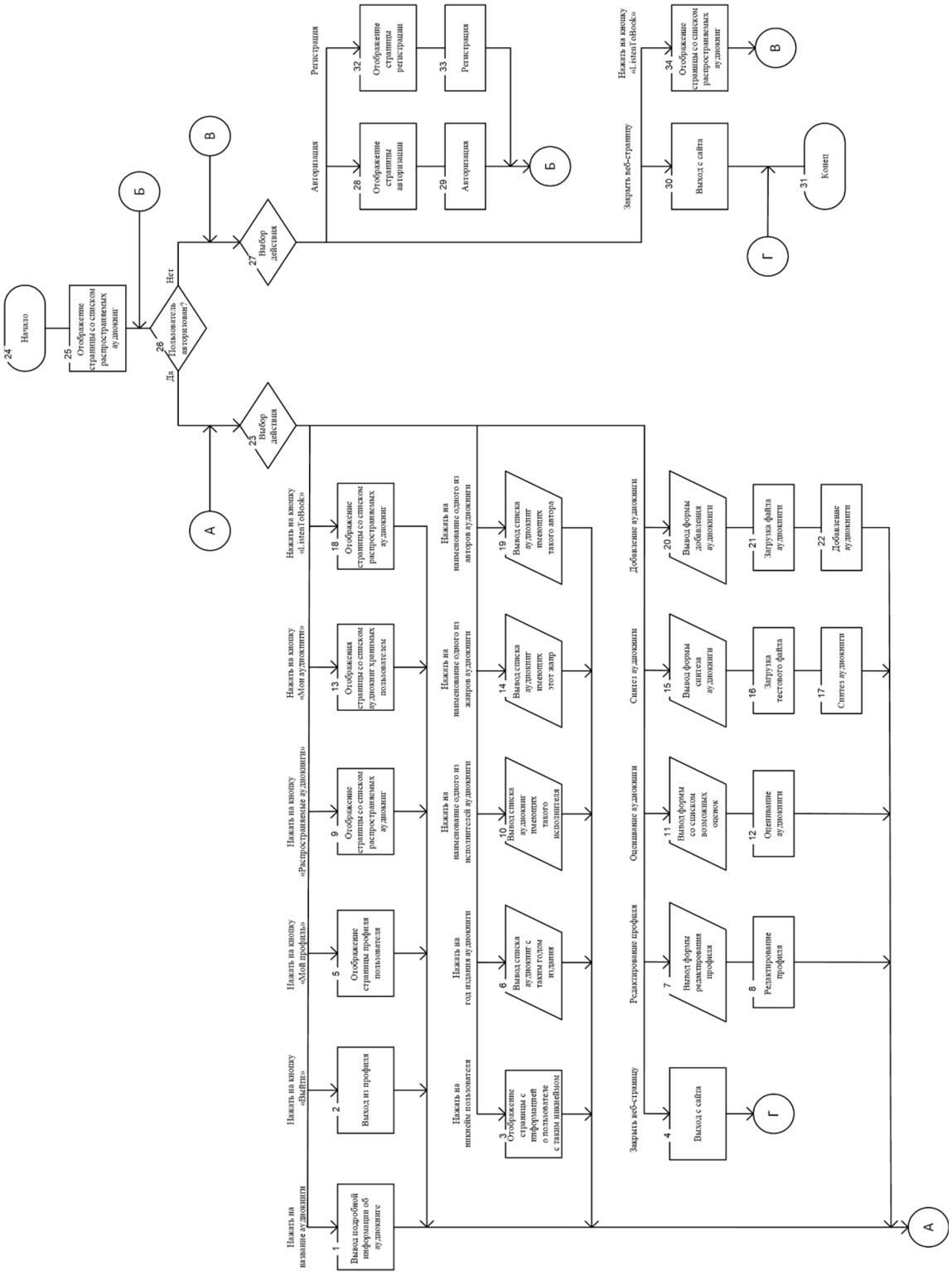

Рисунок 3.3 – Схема программы

На данной схеме представлены все функции приложения, которыми пользователи могу пользоваться в зависимости от своего текущего статуса.

Хранить и оценивать аудиокниги может только зарегистрированный пользователь, поэтому функция регистрации очень важна. На рисунке 3.4 представлен алгоритм регистрации пользователя. Для того чтобы зарегистрировать пользователя необходимо заполнить обязательные поля формы корректными данными и нажать на кнопку «Зарегистрироваться».

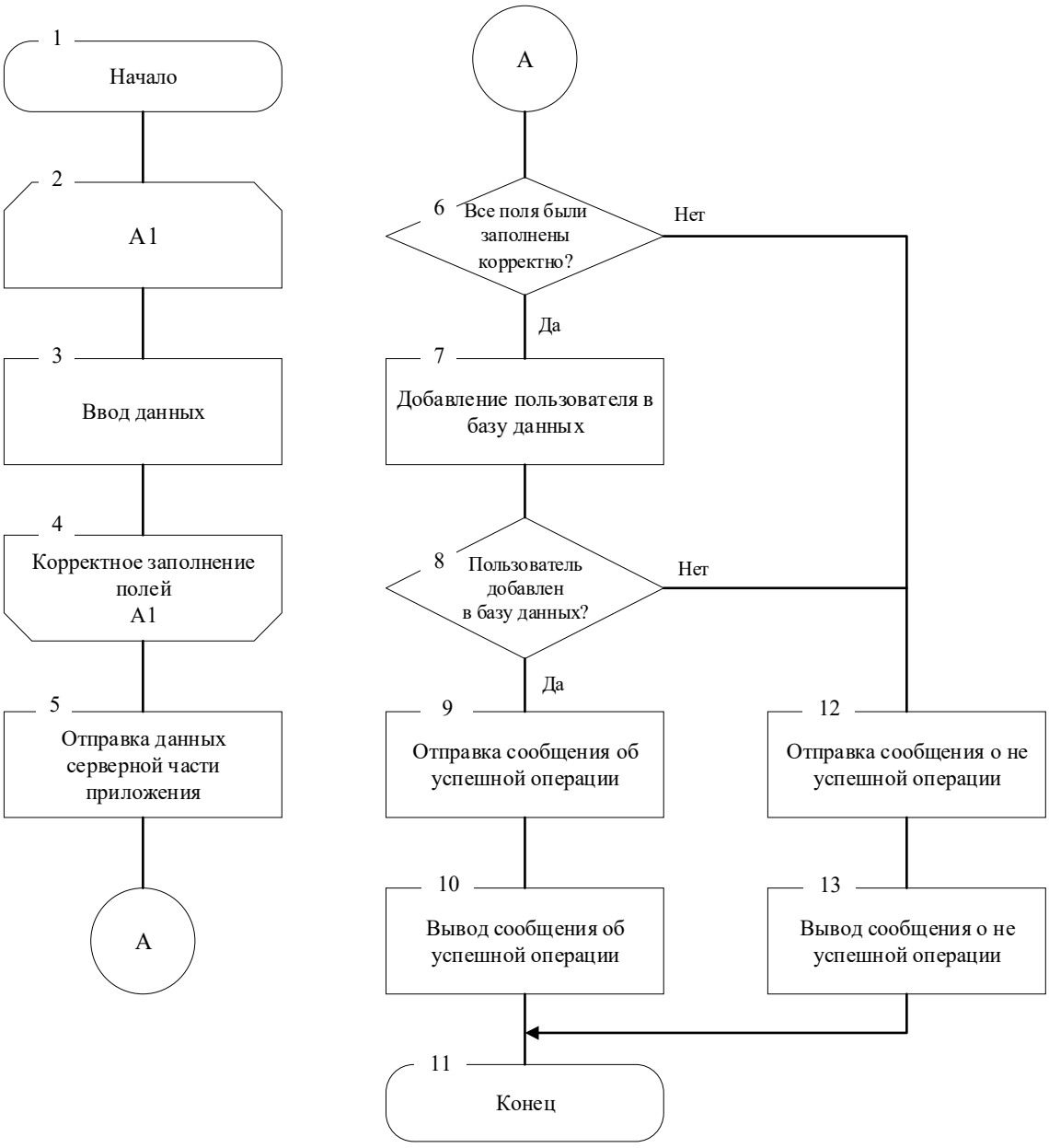

Рисунок 3.4 – Алгоритм регистрации пользователя

Сразу после успешной регистрации пользователь может авторизоваться на сайте. Для авторизации пользователя требуется заполнить обязательные поля формы и нажать на кнопку «Войти» (рисунок 3.5). В случае успешной

авторизации на сайте, пользователь сможет начать использование приложения согласно своей роли.

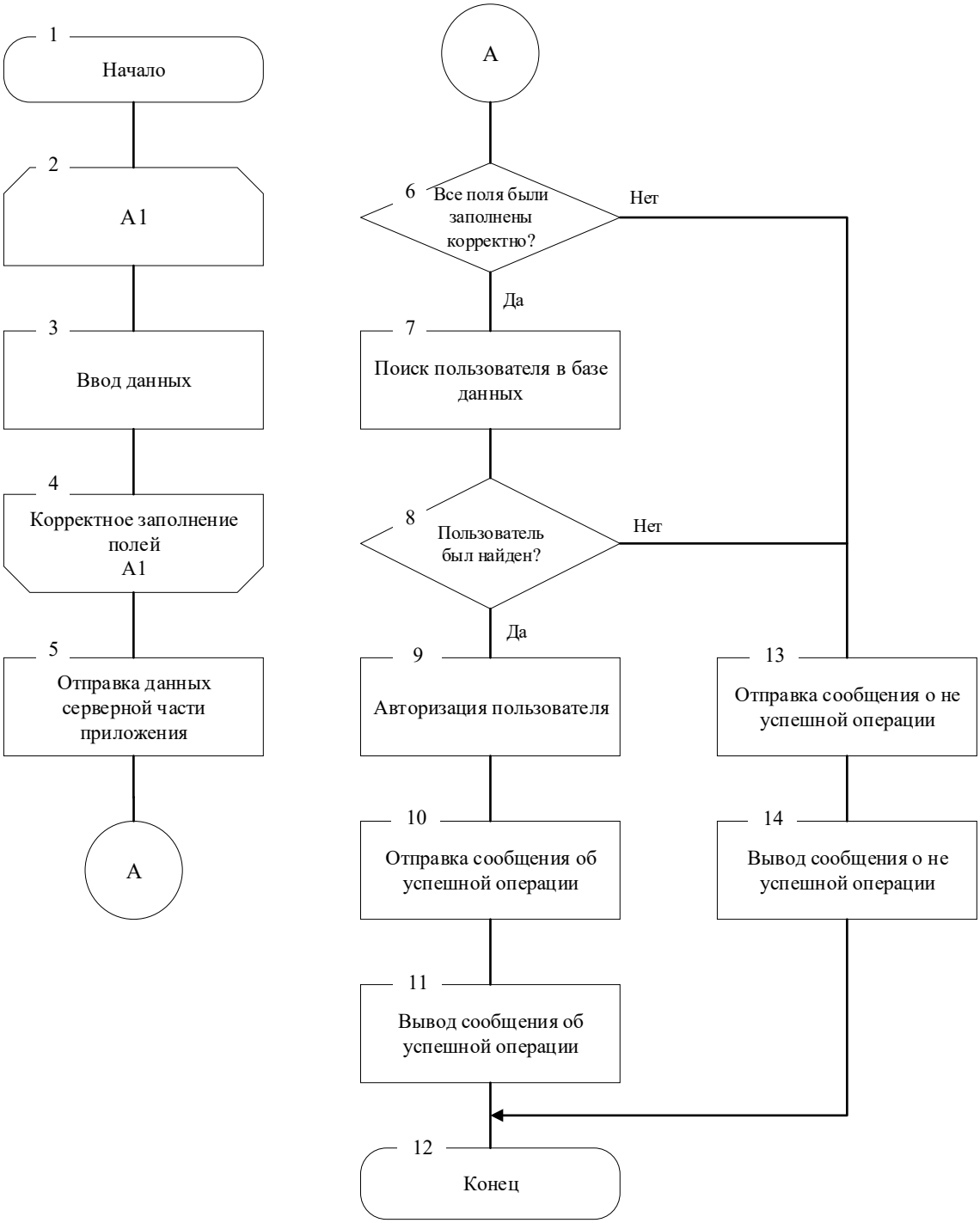

Рисунок 3.5 – Алгоритм авторизации пользователя

Пользователь может изменять данные своего профиля. Алгоритм редактирования профиля пользователя приведён на рисунке 3.6. Чтобы отредактировать данные, пользователь должен изменить значения полей и нажать на

кнопку «Редактировать профиль». Если введённые данные будут некорректны, появится сообщение об ошибке.

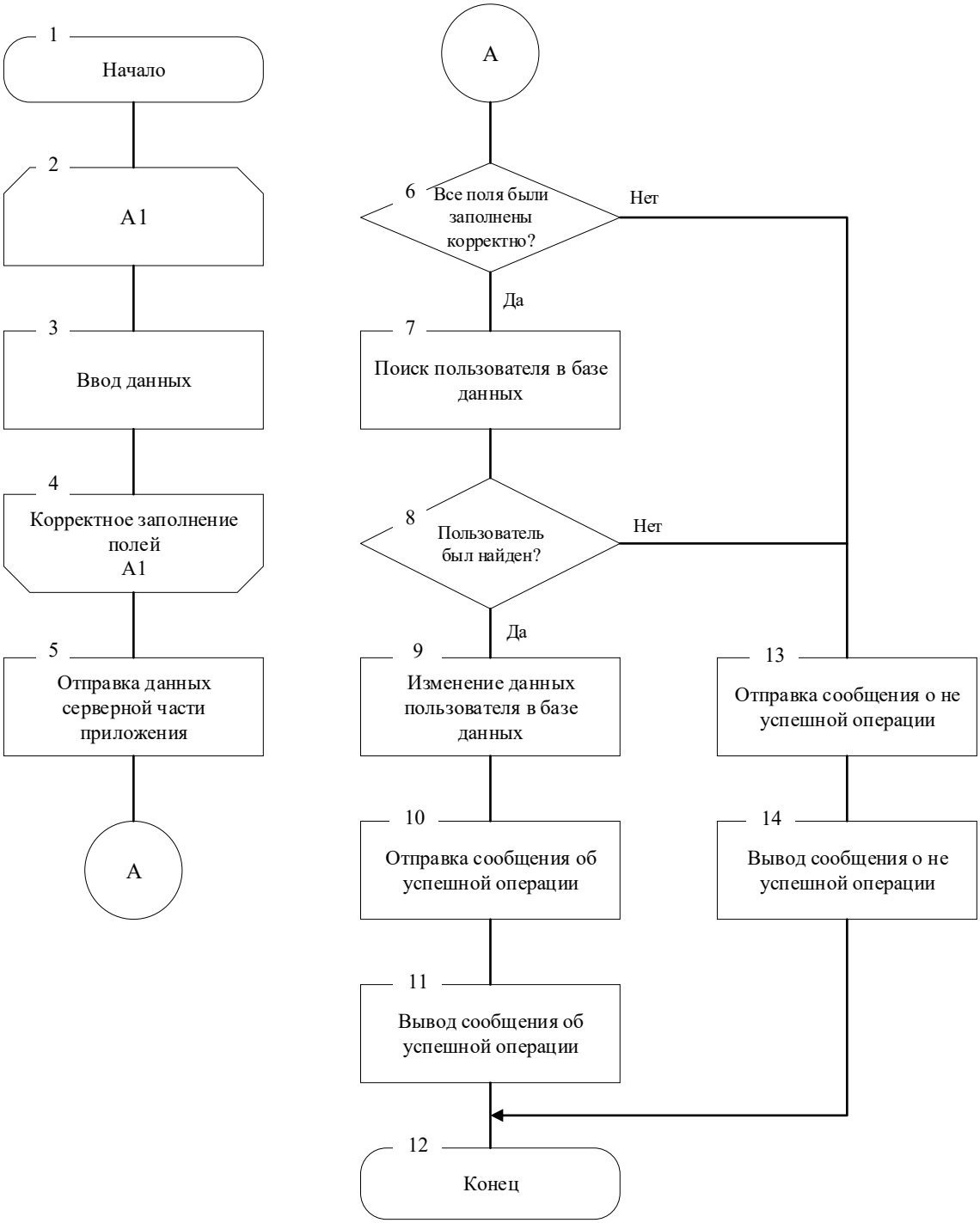

Рисунок 3.6 – Алгоритм редактирования профиля пользователя

Пользователь может добавить, удалить, получить или изменить аудиокнигу. Для этого в приложении используется контроллер аудиокниг. Алгоритм работы контроллера аудиокниг приведён на рисунке 3.7.

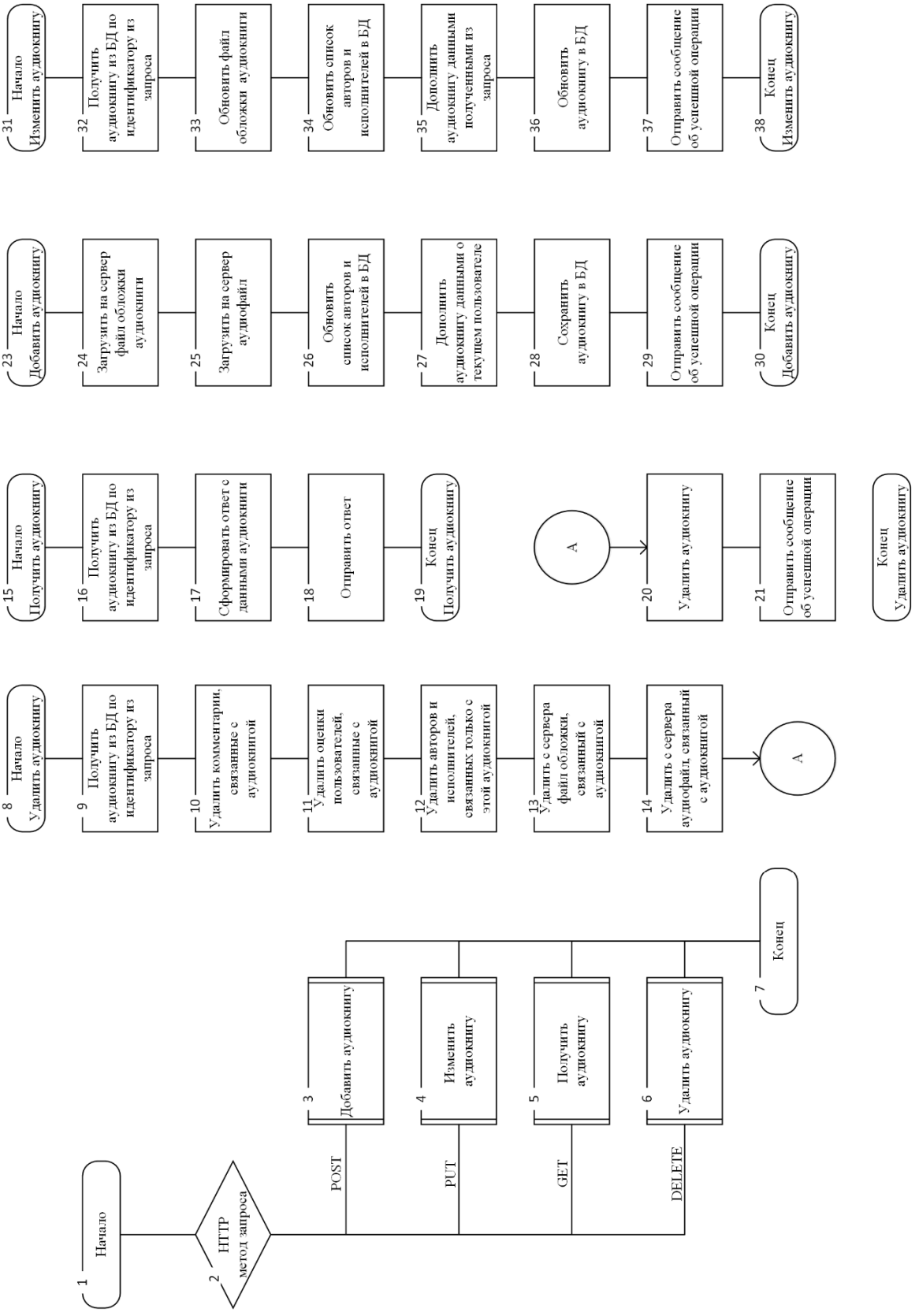

Рисунок 3.7 – Алгоритм работы контроллера аудиокниг

Итак, выше были приведены некоторые алгоритмы разрабатываемого приложения. Согласно этим алгоритмам будет реализована данная функциональность.

4 РАЗРАБОТКА ПРИЛОЖЕНИЯ

4.1 Язык программирования Java

Для написания программного средства был выбран язык программирования Java. На сегодняшний момент данный язык программирования является одним из самых мощных, быстро развивающихся и востребованных языков в ИТ-отрасли. Существует множество приложений и веб-сайтов, которые не работают при отсутствии установленной Java, и с каждым днем число таких веб-сайтов и приложений увеличивается.

Java относится к семье языков с С-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и C#. Java представляет собой язык программирования и платформу вычислений, которая была впервые выпущена Sun Microsystems в 1995 г.

Java отличается быстротой, высоким уровнем защиты и надежностью. В настоящий момент на нем пишутся самые различные приложения: от небольших десктопных программ до крупных веб-порталов и веб-сервисов, обслуживающих ежедневно миллионы пользователей. Приложения, написанные на языке Java, можно обнаружить на любом устройстве от портативных компьютеров и игровых консолей до центров данных и суперкомпьютеров, используемых для научных разработок.

Основные достоинства Java:

Переносимость. Программы, написанные на языке Java, после однократной трансляции в байт-код могут быть исполнены на любой платформе, для которой реализована виртуальная Java-машина.

Безопасность. Функционирование программы полностью определяется (и ограничивается) виртуальной Java-машиной. Отсутствуют указатели и другие механизмы для непосредственной работы с физической памятью и прочим аппаратным обеспечением компьютера.

Надежность. В языке Java отсутствуют механизмы, потенциально приводящие к ошибкам: арифметика указателей, неявное преобразование типов с потерей точности и т.п. Присутствует строгий контроль типов, обязательный контроль исключительных ситуаций. Многие логические ошибки обнаруживаются на этапе компиляции.

Сборщик мусора. Освобождение памяти при работе программы осуществляется автоматически с помощью «сборщика мусора», поэтому программировать с использованием динамически распределяемой памяти проще и надежнее.

Стандартные библиотеки. Многие задачи, встречающиеся при разработке программного обеспечения, уже решены в рамках стандартных библиотек. Использование объектно-ориентированного подхода позволяет легко использовать готовые объекты в своих программах. Для запуска приложения необходима установка JRE, содержащего полный набор библиотек, даже если все они не используются в приложении [14].

MySql Workbench предоставляет:

– Возможность наглядно представить модель базы данных в графическом виде.

– Наглядный и функциональный механизм установки связей между таблицами, в том числе «многие ко многим» с созданием таблицы связей.

– Reverse Engineering — восстановление структуры таблиц из уже существующей на сервере БД.

– Удобный редактор SQL запросов, позволяющий сразу же отправлять их серверу и получать ответ в виде таблицы.

– Возможность редактирования данных в визуальном режиме [15].

4.3 Основные компоненты программного средства

При проектировании программного средства особое внимание уделяется стандартным компонентам Spring Framework, существующим для разработки веб-приложений.

Контроллеры – часть MVC архитектуры содержащаяся в Spring. Они объявляются с помощью аннотации @Controller, все классы, объявленные с данной аннотацией считаются контроллерами. Методы данного вида контроллеров выбираются на основании HTTP метода пришедшего запроса. Это позволяет логически разделить методы на группы для упрощения понимания кода. Контроллеры принято создавать по подходу CRUD (Create-Read-Update-Delete). В соответствии с ним методы контроллера делятся на 4 группы:

– добавление новых данных в приложение (Create);

– извлечение данных из приложения (Read);

– изменение существующих данных (Update);

– удаление данных из приложения (Delete).

За каждую группу методов отвечает свой HTTP метод, используемый для запроса действия. За добавление данных отвечает HTTP метод Post, извлечение – Get, изменение – Put, удаление – Delete. На рисунке 4.2 приведён пример контроллера, отвечающего за работу с пользователем.

5 ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ

Одним из важнейших этапов разработки программного обеспечения является тестирование. Проведение тестирования позволяет повысить качество конечного продукта, а также выполнить проверку работоспособности разрабатываемого программного средства.

Методы тестирования программного обеспечения подразделяются на структурные и функциональные. Структурное тестирование основывается на детальном изучении логики программы и подборе тестов, обеспечивающих максимально возможное число проверяемых операторов, логических ветвлений и условий. Данный вид тестирования находит применение на более ранних этапах разработки программного средства. Наиболее распространены следующие виды структурного тестирования:

1) по критерию:
   – путей;
   – ветвей;
2) базовых путей;
3) циклов.

Функциональное тестирование является основным видом тестирования ПО. Каждая функция программы тестируется (проверяется на правильность в некоторых точках) и при этом делается вывод об ее правильности. Функциональное тестирование не является альтернативой структурному тестированию, поскольку позволяет обнаружить другие классы ошибок:

– некорректные или отсутствующие функции;

– ошибки интерфейса;

– неточности во внешних структурах данных.

В отличие от структурного, функциональное тестирование используется на более поздних этапах тестирования и может выполняться вручную или с использованием средств автоматизации. Целями функционального тестирования являются:

– поиск в тестируемом программном обеспечении ошибок;

– документирование найденных ошибок с целью дальнейшего их исправления;

– определение соответствия тестируемого программного продукта предъявляемым к нему требованиям и принятие объективного заключения о возможности поставки протестированного ПО заказчику.

В силу рассмотренных выше видов тестирования, для проведения тестирования программного средства, в том числе для оценки его работоспособности, было выбрано функциональное тестирование программного средства. Для его проведения был разработан ряд тестовых сценариев. Тестирование программного средства производилась на персональном компьютере с установленной операционной системой Windows 10.

Разработанные текстовые сценарии для пользователя, который не авторизован в программном средстве, представлены в таблице 5.1

Таблица 5.1 – Тестовые сценарии неавторизованного пользователя

Тестируемая функциональность	Последовательность действий	Ожидаемый результат	Полученный результат
1	2	3	4
Просмотр списка распространяемых аудиокниг.	1. Открыть главную страницу приложения.	Отображение страницы со списком распространяемых аудиокниг.	Тест успешно пройден.
Сортировка списка распространяемых аудиокниг.	1. Открыть главную страницу приложения. 2. Выбрать параметры для сортировки. 3. Нажать на кнопку «Сортировать».	Отображение отсортированного списка распространяемых аудиокниг.	Тест успешно пройден.
Фильтрация списка распространяемых аудиокниг.	1. Открыть главную страницу программного средства. 2. Нажать на один из доступных параметров фильтрации.	Отображение отфильтрованного списка распространяемых аудиокниг.	Тест успешно пройден.
Просмотр подробной информации об аудиокниге.	1. Открыть главную страницу программного средства. 2. Нажать на название одной из аудиокниг в списке.	Отображение страницы с подробной информацией о выбранной аудиокниге.	Тест успешно пройден.
Скачивание аудиокниги.	1. Открыть главную страницу программного средства. 2. Нажать на название одной из аудиокниг в списке. 3. Нажать на кнопку «Скачать».	Начало загрузки файла аудиокниги на устройство пользователя.	Тест успешно пройден.

Разработанные тестовые сценарии регистрации и авторизации представлены в таблице 5.2.

Таблица 5.2 – Тестовые сценарии регистрации и авторизации

Тестируемая функциональность	Последовательность действий	Ожидаемый результат	Полученный результат
1	2	3	4
Регистрация.	1. Открыть главную страницу приложения. 2. Выбрать пункт «Регистрация». 3. Заполнить все поля формы корректными данными. 4. Нажать кнопку «Зарегистрироваться».	Отображение страницы авторизации.	Тест успешно пройден.
Регистрация. Валидация ошибок.	1. Открыть главную страницу приложения. 2. Выбрать пункт «Регистрация». 3. Заполнить форму некорректными данными. 4. Нажать кнопку «Зарегистрироваться».	Отображение сообщения о некорректности введённых данных.	Тест успешно пройден.
Авторизация.	1. Открыть главную страницу приложения. 2. Выбрать пункт «Авторизация». 3. Ввести корректные никнейм и пароль в соответствующие поля. 4. Нажать кнопку «Войти».	Авторизация пользователя. Отображение страницы со списком распространяемых аудиокниг.	Тест успешно пройден.
Авторизация. Валидация ошибок.	1. Открыть главную страницу приложения. 2. Выбрать пункт «Авторизация». 3. Ввести некорректные никнейм и пароль в соответствующие поля. 4. Нажать кнопку «Войти».	Отображение сообщения о некорректности введённых данных.	Тест успешно пройден.

Последующие тестовые сценарии требует предварительной авторизации для их успешного выполнения. Разработанные тестовые сценарии, требующие прохождения пользователем авторизации, доступны к рассмотрению в таблице 5.3.

Таблица 5.3 – Тестовые сценарии авторизованного пользователя

Тестируемая функциональность	Последовательность действий	Ожидаемый результат	Полученный результат
1	2	3	4
Просмотр профиля.	1. Открыть главную страницу приложения. 2. Нажать на никнейм пользователя. 3. Выбрать пункт «Мой профиль».	Отображение страницы с информацией о профиле пользователя.	Тест успешно пройден.
Просмотр списка аудиокниг, хранимых пользователем.	1. Открыть главную страницу приложения. 2. Нажать на никнейм пользователя. 3. Выбрать пункт «Мои аудиокниги».	Отображение страницы со списком аудиокниг, хранимых пользователем.	Тест успешно пройден.
Отправка запроса на внесение аудиокниги в список распространяемых.	1. Открыть главную страницу приложения. 2. Нажать на никнейм пользователя. 3. Выбрать пункт «Мои аудиокниги». 4. Нажать на название одной из аудиокниг в списке. 5. Нажать на кнопку «Распространить».	Отображение сообщения об удачной отправке запроса на внесение аудиокниги в список распространяемых.	Тест успешно пройден.
Добавление аудиокниги в список хранимых аудиокниг.	1. Открыть главную страницу приложения. 2. Нажать на кнопку «Добавить аудиокнигу». 3. Заполнить все поля формы корректными данными. 4. Нажать на кнопку «Добавить», расположенную внизу страницы.	Отображение страницы с обновленным списком хранимых аудиокниг.	Тест успешно пройден.

Продолжение таблицы 5.3

1	2	3	4
Добавление аудиокниги в список хранимых аудиокниг. Валидация ошибок.	1. Открыть главную страницу приложения. 2. Нажать на кнопку «Добавить аудиокнигу». 3. Заполнить поля формы некорректными данными. 4. Нажать на расположенную внизу страницы кнопку «Добавить».	Отображение сообщения о некорректности введённых данных.	Тест успешно пройден.
Удаление аудиокниги из списка хранимых книг.	1. Открыть главную страницу приложения. 2. Нажать на никнейм пользователя. 3. Выбрать пункт «Мои аудиокниги». 4. Нажать на название одной из аудиокниг в списке. 5. Нажать на кнопку «Удалить».	Отображение страницы с обновленным списком хранимых аудиокниг.	Тест успешно пройден.
Оценивание аудиокниги.	1. Открыть главную страницу приложения. 2. Нажать на кнопку «Оценить». 3. Выбрать соответствующее значение оценки.	Отображение обновленного рейтинга аудиокниги.	Тест успешно пройден.
Комментирование аудиокниги.	1. Открыть главную страницу программного средства. 2. Нажать на название одной из аудиокниг в списке. 3. Написать текст комментария в соответствующем поле. 4. Нажать на кнопку «Добавить комментарий»	Отображение обновленного списка комментариев аудиокниги.	Тест успешно пройден.
Редактирование профиля.	1. Открыть главную страницу приложения. 2. Нажать на никнейм пользователя. 3. Выбрать пункт «Мой профиль». 4. Нажать на кнопку «Редактировать».	Отображение страницы с обновленной информацией о профиле.	Тест успешно пройден.

Продолжение таблицы 5.3

2	3	4	5
	3. Изменить значения тех полей формы, которые требуется отредактировать. 4. Нажать на кнопку «Сохранить изменения».		
Синтез аудиокниги.	1. Открыть главную страницу приложения. 2. Выбрать пункт «Синтезировать аудиокнигу». 3. Выбрать параметры для синтеза. 4. Загрузить текстовый файл книги. 5. Нажать на кнопку «Начать синтез».	Отображение кнопки для скачивания синтезированной аудиокниги.	Тест успешно пройден.

Были проверены основные функциональные возможности приложения. Все тесты были пройдены успешны, что свидетельствует о том, что приложение успешно справилось с функциональными испытаниями.

6 РУКОВОДСТВО ПО УСТАНОВКЕ И ИСПОЛЬЗОВАНИЮ ПРИЛОЖЕНИЯ

6.1 Установка приложения

В данном разделе приведены основные сведения по работе с программным средством.

Для корректной работы сервера данного приложения необходимо компьютерное устройство с установленной ОС Windows Server 2019 (версии 10 или выше). Также требуется наличие установленного в операционной системе сервера приложений «Apache Tomcat» (версии 10.0.6 или выше). Дополнительно необходимо наличие установленной в операционной системе СУБД MySQL (версии 4.0 или выше).

При развертывании веб-сервера следует следовать актуальным инструкция, размещенным на официальном сайте компании Microsoft в разделе «Документация», а также на официальном сайте компании Apache в раздел «Documentation». Подобный подход обеспечит использование наиболее проверенных и отработанных методик по запуску веб-сервера, а также квалифицированную и оперативную консультативную помощь от сотрудников компании-разработчика.

Клиентская часть приложения, разработанного в рамках данного дипломного проекта не требует установки и настройки на конечных устройствах пользователя, поскольку является веб-приложением.

Для корректной работы приложения необходим один из следующих браузеров с соответствующей минимальной версией:

– Google Chrome 70;
– Mozilla Firefox 66;
– Microsoft Edge 44.

6.2 Руководство по использованию приложения

Для того, чтобы использовать приложение, необходимо открыть браузер и в строку поиска ввести адрес http://local.ListenToBook.com. При первом посещении сайта или после выхода из профиля происходит отображение главной страницы: страницы со списком распространяемых аудиокниг. Интерфейс данной страницы представлен на рисунке 6.1. Здесь пользователь имеет возможность:

– просмотреть список распространяемых аудиокниг, с краткой информацией о каждой;

– отсортировать список аудиокниг, путём выбора критериев сортировки, и нажатия на кнопку «Сортировать»;

– отфильтровать список аудиокниг, введя часть названия книги в строку поиска и нажав на кнопку «Найти»;

7 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ СИНТЕЗА, ХРАНЕНИЯ И РАСПРОСТРАНЕНИЯ АУДИОКНИГ

7.1 Назначение и функции веб-приложения, характеристика пользователей

Разработанное в дипломном проекте веб-приложения предназначено для хранения и распространения аудиокниг в сети интернет, а также их синтеза на основе текста книги.

Применение веб-приложения позволит его пользователям решать следующие задачи:

1) удалённое хранение аудиокниг;
2) получение доступа к крупной базе аудиокниг;
3) поиск и фильтрация аудиокниг по выбранным критериям;
4) озвучивание текстовой информации;
5) синтез аудиокниги по текстовому файлу;
6) прослушивания аудиокниги в браузере;
7) распространение новых аудиокниг в сети интернет.

Функции веб-приложения:

1) хранение аудиокниг, добавленных пользователем;
2) хранение и отображение подробной информации об аудиокниге;
3) предоставление возможности скачивать аудиокниги;
4) предоставление возможности добавить аудиокнигу в список распространяемых;
5) предоставление пользователям сети интернет доступа к списку распространяемых аудиокниг;
6) предоставление возможности поиска аудиокниг по выбранным критериям;
7) предоставление возможности фильтрации списка аудиокниг;
8) предоставление возможности регистрации пользователя;
9) предоставление возможности аутентификации пользователя;
10) предоставление возможности добавления новой аудиокниги в список хранимых пользователем;
11) предоставление возможности синтеза аудиокниги по текстовому файлу;
12) предоставление возможности озвучивания любого печатного текста.

Приложение будет размещено в сети Internet. Компания разработчик получит экономический эффект в виде прироста прибыли от сотрудничества с различными компаниями и сервисами, заинтересованными в размещении рекламных объявление на сайте.

Основными пользователями веб-приложения для синтеза, хранения и

распространения аудиокниг являются предприятия и организации, занятые в сферах социальных услуг и культурного досуга, например:

1) РУП издательство “Беларусь”, местоположение - Республика Беларусь, г. Минск;

2) ГУ "Специальная городская библиотека для слепых", местоположение - Беларусь, г. Минск;

3) Общественное объединение "Белорусское товарищество инвалидов по зрению”, местоположение - Республика Беларусь, г. Минск.

7.2 Расчет затрат на разработку веб-приложения для синтеза, хранения и распространения аудиокниг

7.2.1 Расчёт основной заработной платы участников команды осуществляется по формуле:

$$З_о = К_{пр} \sum_{i=1}^{n} З_{ч.i} t_i , \quad (7.1)$$

где n – количество исполнителей, занятых разработкой конкретного ПО;

$К_{пр}$ – коэффициент премий, равный 1,2;

$З_{ч.i}$ – часовая заработная плата i-го исполнителя, р.;

t_i – трудоёмкость работ, выполняемых i-м исполнителем, ч.

Данные о величине месячной заработной платы различных специалистов в 2020 году получены на ресурсе it-academy.by. При этом принято, что среднее количество рабочих часов в месяце равно 168 ч.

Исходя из данных указанных выше, рассчитаем затраты на основную заработную плату, используя формулу (7.1). Результаты вычислений приведены в таблице 7.1.

Таблица 7.1 – Расчёт затрат на основную заработную плату разработчиков

Наименование должности разработчика	Вид выполняемой работы	Месячная заработная плата, р.	Часовая заработная плата, р.	Трудоёмкость, ч	Зарплата по тарифу, р.
1. Бизнес-аналитик	Формирование технического задания	1800	10,71	62	664,02

Продолжение таблицы 7.1

Наименование должности разработчика	Вид выполняемой работы	Месячная заработная плата, р.	Часовая заработная плата, р.	Трудоёмкость, ч	Зарплата по тарифу, р.
2. Администратор баз данных	Выработка требований к базе данных и её проектирование	1000	5,95	54	321,3
3. Инженер-программист	Разработка алгоритмов работы системы, написание серверной части веб-приложения	2500	14,88	160	2380,8
4. Программист	Разработка клиентской части веб-приложения	1100	6,55	87	569,85
5. Специалист по тестированию программного обеспечения	Тестирование работы веб-приложения	1600	9,52	96	913,92
6. Дизайнер	Разработка дизайна веб-приложения	1100	6,55	64	419,2
Премия (20%)					1053,89
Всего основная заработная плата разработчиков					6322,91

7.2.2 Расчёт дополнительной заработной платы включая выплаты, предусмотренные законодательством о труде, проводиться по формуле:

$$З_д = \frac{З_о \cdot Н_д}{100\%}, \qquad (7.2)$$

где $З_о$ – затраты на основную заработную плату, р.;
$Н_д$ – норматив дополнительной заработной платы равный 20%.

Подставив значения в формулу (7.2), получаем:

$$З_д = \frac{6322{,}91 \cdot 20\%}{100\%} = 1264{,}58\,\text{р}.$$

7.2.3 Отчисления на социальные нужды (в фонд социальной защиты населения и на обязательное страхование) определяются в соответствии с действующими законодательными актами по формуле:

$$Р_{соц} = \frac{(З_о + З_д) \cdot Н_{соц}}{100\%}, \quad (7.3)$$

где $Н_{соц}$ – норматив отчислений на социальные нужды, равный (34 + 0,6)%.

Таким образом, согласно формуле (7.3) размер отчислений в фонд социальной защиты населения и на обязательное страхование составит:

$$Р_{соц} = \frac{(6322,91+1264,58) \cdot (34+0,6)\%}{100\%} = 2625,27 \text{ р}.$$

7.2.4 Прочие затраты:

Для веб-ориентированного приложения пункт «Прочие затраты» включает в себя:

1) Оплату электроэнергии, потребляемой при разработке и тестировании веб-приложения;

2) Оплату аренды рабочих помещений;

3) Стоимость лицензионного программного обеспечения необходимого для разработки веб-приложения;

4) Затраты на оборудование рабочих мест.

Вычисляются прочие затраты как процент от затрат на основную заработную плату разработчиков по формуле:

$$Р_{пз} = \frac{З_о \cdot Н_{пз}}{100\%}, \quad (7.4)$$

где $Н_{пз}$ – норматив прочих затрат, равный 125%.

Соответственно подставив данные в формулу (7.4), получаем:

$$Р_{пз} = \frac{6322,91 \cdot 125\%}{100\%} = 7903,64 \text{ р},$$

7.2.5 Полная сумма затрат на разработку приложения находится путём суммирования всех рассчитанных статей затрат. Все результаты расчётов приведены в таблице 7.2.

Таблица 7.2 – Затраты на разработку программного обеспечения

Наименование статьи затрат	Сумма, р.
1. Основная заработная плата разработчиков	6322,91
2. Дополнительная заработная плата разработчиков	1264,58
3. Отчисления на социальные нужды	2625,27
4. Прочие затраты	7903,64
Общая сумма затрат на разработку	18 116,4

7.3 Оценка эффекта от использования веб-приложения

Экономический эффект от разрабатываемого веб-приложения представляет собой прибыль, полученную от соглашений с компаниями и сервисами, проявившими желание разместить объявление и получившими одобрение от администрации веб-приложения. За год планируется заключить не менее 20 соглашений. Средний уровень цен на размещение объявлений на подобных сайтах составляет 400р, в месяц. Так как соглашение заключается в среднем на 6 месяцев, по итогу получатся 120 месячных платежей.

Так как предприятие является плательщиком налога на прибыль, то итоговая прибыль рассчитывается по формуле:

$$П_{ч} = (Ц \cdot N - НДС - З_{р}) \cdot \left(1 - \frac{Н_{п}}{100\%}\right), \quad (7.5)$$

где $Н_{п}$ – ставка налога на прибыль предприятия, равная 5%;
Ц – средняя величина одного месячного платежа за размещение рекламных объявлений га сайте, р.;
N – количество платежей за год;
$З_{р}$ – затраты на разработку и реализацию программного средства, р.;
НДС – сумма налога на добавленную стоимость, р.

НДС рассчитывается по формуле:

$$НДС = \frac{Ц \cdot N \cdot Н_{дс}}{100\% + Н_{дс}}, \quad (7.6)$$

где $Н_{дс}$ – ставка налога на добавленную стоимость, согласно действующему законодательству равная 20%.

Подставив данные в формулу (7.6), получаем сумму налога на добавленную стоимость:

$$НДС = \frac{400 \cdot 120 \cdot 20\%}{100\% + 20\%} = 8000\,р.$$

Подставив данные в формулу (7.7), определяем прибыль, полученную разработчиком:

$$П_ч = (400 \cdot 120 - 8000 - 18116{,}4) \cdot \left(1 - \frac{5\%}{100\%}\right) = 20789{,}42\,р.$$

Для оценки эффективности затрат на разработку приложения необходимо рассчитать уровень рентабельности затрат по следующей формуле:

$$У_р = \frac{П_ч}{З_р} \cdot 100\%, \qquad (7.7)$$

Подставив значения в формулу (7.7), получаем:

$$У_р = \frac{20789{,}42}{18116{,}4} \cdot 100\% = 114{,}75\%.$$

Проект будет экономически эффективным, если рентабельность затрат на разработку приложения будет не меньше средней процентной ставки по банковским депозитным вкладам. Средняя процентная ставка по банковским вкладам для юридических лиц за февраль 2021 года составила 10,57 %. Исходя из этих данных можно сделать вывод, что проект рентабелен, так как уровень рентабельности затрат на разработку приложения составил 114,75%.

Чтобы определить точный срок окупаемости воспользуемся формулой:

$$С_о = \frac{З_р \cdot Д_п}{П_ч}, \qquad (7.8)$$

где $Д_п$ – срок расчётного периода, равный 12 месяцам.

Соответственно подставив значения в формулу (7.8), получаем срок окупаемости в месяцах:

$$С_о = \frac{18116{,}4 \cdot 12}{20789{,}42} = 10{,}45\,мес.$$

Таким образом, для того, чтобы полностью окупить приложение, потребуется менее 11 месяцев.

7.4 Расчёт показателей эффективности инвестиций в разработку приложения

Для определения эффективности инвестиций в разработку веб-приложения, необходимо сравнить размер инвестиций (затраты на разработку ПО), и получаемый годовой экономический эффект.

Экономическую эффективность инвестирования в разработку данного веб-приложения можно отобразить через рентабельность инвестиций, которая вычисляется по формуле:

$$Р_{и} = \frac{П_{ч}}{З_{р}} \cdot 100\%. \quad (7.9)$$

Подставляя значения в формулу (7.9), получаем процент рентабельности инвестиций:

$$Р_{и} = \frac{20789,42}{18116,4} \cdot 100\% = 114,75\%.$$

В данном случае инвестирование в разработку целесообразно так как рентабельность инвестиций превышает 110,57% (100% плюс ставка по банковским депозитам), из чего следует что вложится в проект выгоднее чем внести деньги на банковский депозит.

7.5 Вывод

Таким образом, при составлении технико-экономического обоснования были определены зарплаты по тарифу для каждого разработчика, вследствие чего с учётом премии в 25% стало возможным рассчитать основную заработную плату разработчиков, значение которой составило 6322,91р.

После дополнительных расчётов, были определены: затраты на дополнительную заработную плату разработчиков равные 1264,58р., отчисления на социальные нужды, составившие 2625,27р., и прочие затраты равные 7903,64р. И в итоге, была рассчитана общая сумма затрат на разработку веб-приложения равная 18 116,4р.

Далее, после определения экономического эффекта получаемого от разработки веб-приложения, была подсчитана прибыль за первый год равная 20789,42р., которая превышает общую сумму затрат на разработку 18 116,4р. А рассчитанный на их основе уровень, рентабельность равный114,75 %, явно указывает на прибыльность вложений в разработку.

Таким образом, полученные результаты технико-экономического обоснования «Веб-приложения для синтеза, хранения и распространения аудиокниг, на базе Spring Framework» свидетельствуют об экономической эффективности разработки данного веб-приложения. Окупаемость разработки произойдёт в течение 11 месяцев. А выгода инвестирования в разработку данного приложения, выше чем при внесении средств на банковский депозит.

ЗАКЛЮЧЕНИЕ

В ходе работы над дипломным проектом был проведён анализ предметной области, проанализированы литературные источники, а также проведено исследование для выявления существующих аналогов, чтобы выделить их достоинства и недостатки, которые необходимо было устранить в разрабатываемом приложении. По результатам исследования были сформулированы общие требования к создаваемому веб-приложению.

Во время работы проведён этап моделирования приложения, в котором были сформулированы функциональные требования к приложению и составлены полные спецификации к ним.

На основе функциональных требований произведено проектирование приложения, разработана программная архитектура, диаграмма развёртывания программного средства, а также разработаны алгоритмы функций приложения.

Были изучены необходимые библиотеки, фреймворки и шаблоны, требующиеся для создания приложения.

Также для обеспечения стабильной работы приложения разработаны тестовые случаи, покрывающие всю его функциональность. Все тесты успешно выполняются, что свидетельствует о корректном исполнении функций приложения.

На завершающем этапе подробно описана методика использования приложения, позволяющая в короткие сроки освоить работу с ним.

Было проведено технико-экономическое обоснование дипломного проекта, которое показало, что приложение является экономически выгодным, т. к. окупается за приемлемые сроки.

Таким образом, итогом дипломного проектирования стало веб-приложение, которое помогает быстро и дёшево создавать, хранить и распространять аудиокниги. Разработанное программное средство полностью соответствует спецификации требований. Данное приложения значительно повысит доступность литературных произведений для людей с ограниченными возможностями.

Таким образом, все поставленные на данный дипломный проект задачи были успешно выполнены.

Работа над приложением будет продолжаться. Направлениями улучшения разработанного приложения являются: качество и количество информации, хранимой о каждой аудиокниге, что позволит пользователю быстрее находить необходимую литературу, а также добавление в приложение интерфейса для слепых и слабовидящих.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Макконнелл, С. Совершенный код. Мастер-класс / С. Макконнелл. – М.: Издательско-торговый дом «Русская редакция», 2010. – 896 с.

[2] Антопольский А. Б., Правовые и технологические проблемы создания и функционирования электронных библиотек. / А. Б. Антопольский, Т. С. Маркарова, Е. А. Данилина – М.: ИНИЦ «Патент», 2008. — 192 с. — ISBN 978-5-89513-119-0.

[3] Баранова Л. Т., Что такое «аудиокнига» и история ее развития/ Л. Т. Баранова // Актуальные проблемы гуманитарных и естественных наук. – 2015. – №3. – С. 118–121. — ISSN 2073-0071

[4] Фролов А., Синтез и распознавание речи. Современные решения [Электронный ресурс] / Фролов А., Фролов Г. – Электрон. журн. – 2003. – Режим доступа: http://www.frolov-lib.ru/. – Дата доступа: 22.05.21.

[5] Лобанов Б. М., Компьютерный синтез и клонирование речи. / Б. М Лобанов, Л. И. Цирульник – Минск: Издательский дом «Белорусская Наука», 2008. — 316 с.

[6] Audio-knigki.com [Электронный ресурс]. – Электронные данные. – Режим доступа: https://audio-knigki.com/. – Дата доступа: 22.05.2021.

[7] Baza-Knig [Электронный ресурс]. – Электронные данные. – Режим доступа: https://baza-knig.ru/. – Дата доступа: 22.05.21.

[8] Au-books [Электронный ресурс]. – Электронные данные. – Режим доступа: https://au-books.com/. – Дата доступа: 22.05.21.

[9] VoxWorker [Электронный ресурс]. – Электронные данные. – Режим доступа: https://voxworker.com/ru. – Дата доступа: 22.05.21.

[10] UniTools [Электронный ресурс]. – Электронные данные. – Режим доступа: https://unitools.tech/voice. – Дата доступа: 22.05.21.

[11] Zvukogram [Электронный ресурс]. – Электронные данные. – Режим доступа: https://zvukogram.com/. – Дата доступа: 22.05.21.

[12] Рогочев С., Обобщенный Model-View-Controller [Электронный ресурс]. – Электронные данные. – Режим доступа: http://rsdn.org/article/patterns/generic-mvc.xml. – Дата доступа: 22.05.21.

[13] Структура реляционных баз, данных [Электронный ресурс]. – Электронные данные. – Режим доступа: https://www.oracle.com/ru/database/what-is-a-relational-database/ – Дата доступа: 22.05.21.

[14] Шилдт Г., Java 8. Руководство для начинающих / Г. Шилдт – М.: Издательский дом «Вильямс», 2015. — 720 с. ISBN: 978-5-8459-1955-7

[15] Бьюли А., Изучаем SQL. / А. Бьюли – М.: Издательский дом «Символ-Плюс», 2016. — 312 с. ISBN: 978-5-93286-051-9

ПРИЛОЖЕНИЕ А
(обязательное)
Исходный код программы

AudiobooksController.java

```
package org.atsynthesizer.demo.controller;

import org.atsynthesizer.demo.entity.*;
import org.atsynthesizer.demo.service.*;
import org.springframework.beans.factory.annotation.Autowired;

import
org.springframework.boot.autoconfigure.data.web.SpringDataWebPro
perties;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Pageable;
import org.springframework.data.domain.Sort;
import
org.springframework.security.core.annotation.AuthenticationPrinc
ipal;
import
org.springframework.security.core.userdetails.UserDetails;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.util.UriComponentsBuilder;

import java.util.ArrayList;
import java.util.List;
import java.util.Objects;

@Controller
@RequestMapping("audiobooks")
public class AudiobooksController {

    private static int PAGINATION_SIZE = 10;

    private AudiobookService audiobookService;

    @Autowired
    private UserService userService;

    @Autowired
    public void setAudiobookService(AudiobookService
audiobookService) {
        this.audiobookService = audiobookService;
    }
```

```
    @Autowired
    private GenreService genreService;

    @Autowired
    private GenreService queryService;

    @Autowired
    private CreatorService creatorService;

    @RequestMapping(value = "", method = RequestMethod.GET)
    public String showAllAudiobooks(
            @RequestParam(defaultValue = "0") int pageNum,
            @RequestParam(required = false) String title,
            @RequestParam(required = false) Long genre,
            @RequestParam(required = false) Long author,
            @RequestParam(required = false) Long performer,
            @RequestParam(required = false) Long year,
            @RequestParam(required = true, defaultValue =
"addDate") String sortBy,
            @RequestParam(required = true, defaultValue =
"false") Boolean ascending,
            @RequestParam(required = false) Boolean all,
            Model model) {

        Pageable page;
        if (ascending) {
            page = PageRequest.of(pageNum, PAGINATION_SIZE,
Sort.by(sortBy).ascending());
        }
        else
        {
            page = PageRequest.of(pageNum, PAGINATION_SIZE,
Sort.by(sortBy).descending());
        }
        Page<Audiobook> audiobooks;

        UriComponentsBuilder builder =
UriComponentsBuilder.newInstance()
                .scheme("http")
                .host("localhost")
                .port(8088)
                .path("/audiobooks");


        if(!Objects.isNull(title)){
            builder.queryParam("title", title);
            audiobooks =
audiobookService.getAudiobooksByTitle(title, page);
        } else if(!Objects.isNull(genre)){
            builder.queryParam("genre", genre);
            audiobooks =
audiobookService.getAudiobooksByGenre(genreService.getById(genre
```

```
), page);
        } else if(!Objects.isNull(author)){
            builder.queryParam("author", author);
            audiobooks =
audiobookService.getAudiobooksByCreator(creatorService.getById(a
uthor), page);
        } else if(!Objects.isNull(performer)){
            builder.queryParam("performer", performer);
            audiobooks =
audiobookService.getAudiobooksByCreator(creatorService.getById(p
erformer), page);
        } else if(!Objects.isNull(year)){
            builder.queryParam("year", year);
            audiobooks =
audiobookService.getAudiobooksByYear(year, page);
        } else{
            builder.queryParam("all", true);
            audiobooks = audiobookService.allAudiobooks(page);
        }

        String oldUrl = builder.build()
                .toUri()
                .toString();

        model.addAttribute("audiobooksInfos",
audiobooks.getContent());
        model.addAttribute("lastPage",
audiobooks.getTotalPages());
        model.addAttribute("currentPage", pageNum);

        model.addAttribute("sortParam", sortBy);
        model.addAttribute("sortAscending", ascending);
        model.addAttribute("oldUrl", oldUrl);

        return "audiobooksPage";
    }

    @RequestMapping(value = "/user", method = RequestMethod.GET)
    public String showUserAudiobooks(
            @RequestParam(defaultValue = "0") int pageNum,
            @RequestParam(required = false) String title,
            @RequestParam(required = false) Long genre,
            @RequestParam(required = false) Long author,
            @RequestParam(required = false) Long performer,
            @RequestParam(required = false) Long year,
            @RequestParam(required = true, defaultValue =
"addDate") String sortBy,
            @RequestParam(required = true, defaultValue =
"false") Boolean ascending,
            @RequestParam(required = false) Boolean all,
            @AuthenticationPrincipal UserDetails currentUser,
            Model model) {
```

```
        Pageable page;
        if (ascending) {
            page = PageRequest.of(pageNum, PAGINATION_SIZE,
Sort.by(sortBy).ascending());
        }
        else
        {
            page = PageRequest.of(pageNum, PAGINATION_SIZE,
Sort.by(sortBy).descending());
        }
        Page<Audiobook> audiobooks;

        UriComponentsBuilder builder =
UriComponentsBuilder.newInstance()
                .scheme("http")
                .host("localhost")
                .port(8088)
                .path("/audiobooks/user");

        User user =
userService.getByNickname(currentUser.getUsername());

        if(!Objects.isNull(title)){
            builder.queryParam("title", title);
            audiobooks =
audiobookService.getAudiobooksByTitle(user, title, page);
        } else if(!Objects.isNull(genre)){
            builder.queryParam("genre", genre);
            audiobooks =
audiobookService.getAudiobooksByGenre(user,
genreService.getById(genre), page);
        } else if(!Objects.isNull(author)){
            builder.queryParam("author", author);
            audiobooks =
audiobookService.getAudiobooksByCreator(user,
creatorService.getById(author), page);
        } else if(!Objects.isNull(performer)){
            builder.queryParam("performer", performer);
            audiobooks =
audiobookService.getAudiobooksByCreator(user,
creatorService.getById(performer), page);
        } else if(!Objects.isNull(year)){
            builder.queryParam("year", year);
            audiobooks =
audiobookService.getAudiobooksByYear(user, year, page);
        } else{
            builder.queryParam("all", true);
            audiobooks = audiobookService.allAudiobooks(user,
page);
        }
```

```
        String oldUrl = builder.build()
                .toUri()
                .toString();

        model.addAttribute("audiobooksInfos",
audiobooks.getContent());
        model.addAttribute("lastPage",
audiobooks.getTotalPages());
        model.addAttribute("currentPage", pageNum);

        model.addAttribute("sortParam", sortBy);
        model.addAttribute("sortAscending", ascending);
        model.addAttribute("oldUrl", oldUrl);

        return "audiobooksPage";
    }


    @RequestMapping(value = "/edit/{id}", method =
RequestMethod.GET)
    public String getClientEditPage(@PathVariable("id") Long id,
                                    @RequestParam(required =
false) String message, Model model) {
        if(!Objects.isNull(message)) {
            model.addAttribute("message", message);
        }

        Audiobook audiobook = audiobookService.getById(id);
        model.addAttribute("audiobookInfo", audiobook);
        Iterable<Genre> genres = genreService.allGenres();
        model.addAttribute("allGenres", genres);
        Iterable<Creator> creators =
creatorService.allCreators();
        model.addAttribute("allCreators", creators);
        return "audiobookEditPage";
    }


    @RequestMapping(value = "/add", method = RequestMethod.GET)
    public String getClientAddPage(@RequestParam(required =
false) String message, Model model) {
        if(!Objects.isNull(message)) {
            model.addAttribute("message", message);
        }

        Audiobook audiobook = audiobookService.getFirst();
        model.addAttribute("audiobookInfo", audiobook);
        Iterable<Genre> genres = genreService.allGenres();
        model.addAttribute("allGenres", genres);
        Iterable<Creator> creators =
creatorService.allCreators();
        model.addAttribute("allCreators", creators);
```

```
        return "audiobookAddPage";
    }

}
```

UserController.java

```
package org.atsynthesizer.demo.controller;

import org.atsynthesizer.demo.entity.Audiobook;
import org.atsynthesizer.demo.entity.Comment;
import org.atsynthesizer.demo.entity.User;
import org.atsynthesizer.demo.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import
org.springframework.security.core.annotation.AuthenticationPrinc
ipal;
import
org.springframework.security.core.userdetails.UserDetails;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.*;

import javax.validation.Valid;
import java.util.Objects;

@Controller
@RequestMapping("user")
public class UserController {

    @Autowired
    private UserService userService;


    @RequestMapping(value = "/me", method = RequestMethod.GET)
    public String getMyUserPage(Model model,
@AuthenticationPrincipal UserDetails currentUser) {
        User user =
userService.getByNickname(currentUser.getUsername());

        model.addAttribute("userInfo", user);
        return "userPage";
    }

    @RequestMapping(value = "/{id}", method = RequestMethod.GET)
    public String getUserPage(Model model,
@AuthenticationPrincipal UserDetails currentUser,
                              @PathVariable("id") Long id) {
        User user = userService.getById(id);
```

```
        model.addAttribute("userInfo", user);
        return "userPage";
    }

    @RequestMapping(value = "/me/edit", method =
RequestMethod.GET)
    public String getUserEditPage(Model model,
@RequestParam(name = "error", required = false) Byte error,
@AuthenticationPrincipal UserDetails currentUser) {
        User user =
userService.getByNickname(currentUser.getUsername());
        if (!Objects.isNull(error)) {

            switch (error){
                case 1:
                    model.addAttribute("errorMessage", "Пароли
не совпадают");
                    break;
                case 2:
                    model.addAttribute("errorMessage",
"Пользователь с таким именем уже существует");
                    break;
            }
        }
        model.addAttribute("userInfo", user);
        return "userEditPage";
    }


    @RequestMapping(value = "/add", method = RequestMethod.POST)
    public String addUser(@ModelAttribute("userForm") @Valid
User userForm, BindingResult bindingResult, Model model) {

        if (bindingResult.hasErrors()) {
            return "redirect:/registration";
        }
        if
(!userForm.getPassword().equals(userForm.getPasswordConfirm())){
            model.addAttribute("errorMessage", "Пароли не
совпадают");
            return "registrationPage";
        }
        if (!userService.add(userForm)){
            model.addAttribute("errorMessage", "Пользователь с
таким именем уже существует");
            return "registrationPage";
        }

        return "redirect:/login";
    }
```

```
    @RequestMapping(value = "/edit", method = RequestMethod.PUT)
    public String editUser(@RequestBody User userForm) {

        if
(!userForm.getPassword().equals(userForm.getPasswordConfirm())){
            return "redirect:/user/me/edit?error=1";
        }
        if (!userService.edit(userForm)){
            return "redirect:/user/me/edit?error=2";
        }

        return "redirect:/user/me";
    }


}
```

AudiobookController.java

```
package org.atsynthesizer.demo.controller;

import org.apache.commons.lang3.RandomStringUtils;
import org.atsynthesizer.demo.entity.*;
import org.atsynthesizer.demo.service.*;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.data.domain.Sort;
import
org.springframework.security.core.annotation.AuthenticationPrinc
ipal;
import
org.springframework.security.core.userdetails.UserDetails;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.multipart.MultipartFile;

import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.sql.Timestamp;
import java.time.Instant;
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;
```

```
@Controller
@RequestMapping("audiobook")
public class AudiobookController {

    @Value("${upload.path}")
    private String UPLOAD_DIR;

    @Autowired
    private AudiobookService audiobookService;

    @Autowired
    private CommentService commentService;

    @Autowired
    private AudiobookFileService audiobookFileService;

    @Autowired
    private UserService userService;

    @Autowired
    private CreatorService creatorService;

    @Autowired
    public void setAudiobookService(AudiobookService
audiobookService) {
        this.audiobookService = audiobookService;
    }


    @RequestMapping(value = "/edit", method =
RequestMethod.POST)
    public String  addAudiobook(
            @RequestParam(required = false) MultipartFile
picture,
            @RequestParam(required = true) List<String> authors,
            @RequestParam(required = true) List<String>
performers,
            @AuthenticationPrincipal UserDetails currentUser,
            @ModelAttribute("audiobookInfo") Audiobook
audiobook, Model model) throws IOException {

        if((!picture.isEmpty()) && (picture.getSize() != 0)) {

audiobook.setPicturePath(audiobookFileService.saveUploadedFile(p
icture, audiobook.getUser()));
            if (audiobook.getPicturePath().isEmpty()) {
                return "redirect:/audiobooks/add";
            }
        }

        List<Creator> creators = new ArrayList<Creator>();
```

```
        if(!authors.isEmpty()) {
            for (String author: authors) {
                Creator creator = new Creator();
                creator.setAuthor(true);
                creator.setTitle(author);
                creators.add(creatorService.add(creator));
            }
        }
        else{
            return
"redirect:/audiobooks/edit/"+audiobook.getId();
        }
        if(!performers.isEmpty()) {
            for (String performer: performers) {
                Creator creator = new Creator();
                creator.setAuthor(false);
                creator.setTitle(performer);
                creatorService.add(creator);
                creators.add(creatorService.add(creator));
            }
        }
        else{
            return
"redirect:/audiobooks/edit/"+audiobook.getId();
        }
        audiobook.setAudiobookCreators(creators);

        audiobookService.edit(audiobook);

        return "redirect:/audiobook/"+audiobook.getId();
    }


    @RequestMapping(value = "/add", method = RequestMethod.POST)
    public String  addAudiobook(
            @RequestParam(required = false) MultipartFile
picture,
            @RequestParam(required = true) MultipartFile
audiobookFileStream,
            @RequestParam(required = true) List<String> authors,
            @RequestParam(required = true) List<String>
performers,
            @AuthenticationPrincipal UserDetails currentUser,
            @ModelAttribute("audiobookInfo") Audiobook
audiobook, Model model) throws IOException {

        if((!picture.isEmpty()) && (picture.getSize() != 0))  {

audiobook.setPicturePath(audiobookFileService.saveUploadedFile(p
icture, currentUser));
            if (audiobook.getPicturePath().isEmpty()) {
                return "redirect:/audiobooks/add";
```

```
            }
        }
        if((!audiobookFileStream.isEmpty()) &&
(audiobookFileStream.getSize() != 0))  {
            String filePath
=audiobookFileService.saveUploadedFile(audiobookFileStream,
currentUser);
            if (filePath.isEmpty()){
                return "redirect:/audiobooks/add";
            }
            String fileExt =
filePath.substring(filePath.lastIndexOf('.'));
            AudiobookFile audiobookFile = new AudiobookFile();
            audiobookFile.setExtension(fileExt);
            audiobookFile.setFilePath(filePath);

audiobookFile.setSize(audiobookFileService.getFileSize(new File(
UPLOAD_DIR+ filePath)));


audiobook.setAudiobookFile(audiobookFileService.add(audiobookFil
e));
        }
        else{
            return "redirect:/audiobooks/add";
        }
        List<Creator> creators = new ArrayList<Creator>();
        if(!authors.isEmpty()) {
            for (String author: authors) {
                Creator creator = new Creator();
                creator.setAuthor(true);
                creator.setTitle(author);
                creators.add(creatorService.add(creator));
            }
        }
        else{
            return "redirect:/audiobooks/add";
        }
        if(!performers.isEmpty()) {
            for (String performer: performers) {
                Creator creator = new Creator();
                creator.setAuthor(false);
                creator.setTitle(performer);
                creatorService.add(creator);
                creators.add(creatorService.add(creator));
            }
        }
        else{
            return "redirect:/audiobooks/add";
        }
        audiobook.setAudiobookCreators(creators);
```

```
        User user =
userService.getByNickname(currentUser.getUsername());
        audiobook.setUser(user);

        if(user.getRole().getTitle().equals("ROLE_ADMIN")){
            audiobook.setDistributed(true);
        }

        Timestamp ts = Timestamp.from(Instant.now());
        audiobook.setAddDate(ts);
        audiobook.setRating(0d);

        audiobookService.add(audiobook);

        return "redirect:/audiobooks";
    }

    @RequestMapping(value="/delete/{id}", method =
RequestMethod.GET)
    public String deleteAudiobook(@PathVariable("id") Long id) {
        Audiobook audiobook = audiobookService.getById(id);
        audiobookService.delete(audiobook);
        return "redirect:/audiobooks";
    }

    @RequestMapping(value = "/{id}", method = RequestMethod.GET)
    public String showAudiobook(@PathVariable("id") Long id,
Model model) {

        Audiobook audiobook = audiobookService.getById(id);

audiobook.getComments().sort(Comparator.comparing(Comment::getSe
ndDateTime));
        model.addAttribute("audiobookInfo", audiobook);


        Comment  comment= new Comment();
        model.addAttribute("newComment", comment);
        return "audiobookDetailsPage";
    }

}
```

QueryController.java

```
package org.atsynthesizer.demo.controller;

import org.atsynthesizer.demo.entity.Audiobook;
import org.atsynthesizer.demo.entity.Comment;
import org.atsynthesizer.demo.entity.Query;
import org.atsynthesizer.demo.entity.User;
```

```
import org.atsynthesizer.demo.service.AudiobookService;
import org.atsynthesizer.demo.service.CommentService;
import org.atsynthesizer.demo.service.QueryService;
import org.atsynthesizer.demo.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.data.domain.Sort;
import
org.springframework.security.core.annotation.AuthenticationPrinc
ipal;
import
org.springframework.security.core.userdetails.UserDetails;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.util.UriComponentsBuilder;

import java.sql.Timestamp;
import java.time.Instant;
import java.util.List;
import java.util.Objects;

@Controller
@RequestMapping("query")
public class QueryController {

    private static final int PAGINATION_SIZE = 10;

    @Autowired
    private AudiobookService audiobookService;

    @Autowired
    private UserService userService;

    @Autowired
    private QueryService queryService;

    @RequestMapping(value="/{id}/success", method =
RequestMethod.GET)
    public String successQuery(@PathVariable("id") Long id) {
        Query query = queryService.getById(id);

        Audiobook audiobook = query.getAudiobook();
        audiobook.setDistributed(true);
        audiobookService.edit(audiobook);

        queryService.delete(query);
        return "redirect:/query/page";
    }
```

```
    @RequestMapping(value="/delete/{id}", method =
RequestMethod.GET)
    public String deleteQuery(@PathVariable("id") Long id) {
        Query query = queryService.getById(id);
        queryService.delete(query);
        return "redirect:/query/page";
    }

    @RequestMapping(value = "/page", method = RequestMethod.GET)
    public String showAllQuery(
            @RequestParam(defaultValue = "0") int pageNum,
            Model model) {

        Pageable page = PageRequest.of(pageNum, PAGINATION_SIZE,
Sort.by("sendDateTime").ascending());

        Page<Query> queries = queryService.allQuery(page);

        model.addAttribute("queriesInfos",
queries.getContent());
        model.addAttribute("lastPage", queries.getTotalPages());
        model.addAttribute("currentPage", pageNum);

        return "queriesPage";
    }

    @RequestMapping(value = "/add", method = RequestMethod.GET)
    public String addQuery(
            @RequestParam(required = true) Long audiobookId,
            @RequestParam(required = true) String oldUrl,
            @AuthenticationPrincipal UserDetails currentUser,
            Model model) {

        Query query = new Query();
        Audiobook audiobook =
audiobookService.getById(audiobookId);
        User user =
userService.getByNickname(currentUser.getUsername());
        query.setAudiobook(audiobook);
        query.setUser(user);

        Timestamp ts = Timestamp.from(Instant.now());
        query.setSendDateTime(ts);

        if(!queryService.getByUserAndAudiobook(user,
audiobook).isPresent()) {
            queryService.add(query);
        }
        if(oldUrl.isEmpty()){
            return "redirect:/audiobook/"+audiobookId;
        }
```

```
        return "redirect:"+oldUrl;
    }

}
```

SpringSecurityConfig.java

```
package org.atsynthesizer.demo.config;
import org.atsynthesizer.demo.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import
org.springframework.security.config.annotation.authentication.bu
ilders.AuthenticationManagerBuilder;
import
org.springframework.security.config.annotation.web.builders.Http
Security;
import
org.springframework.security.config.annotation.web.configuration
.WebSecurityConfigurerAdapter;
import
org.springframework.security.core.userdetails.UserDetailsService
;
import
org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder
;
import
org.springframework.security.web.access.AccessDeniedHandler;

@Configuration
public class SpringSecurityConfig extends
WebSecurityConfigurerAdapter {

    @Autowired
    private AccessDeniedHandler accessDeniedHandler;

    @Qualifier("userServiceImpl")
    @Autowired
    private UserDetailsService userService;

    @Bean
    public BCryptPasswordEncoder bCryptPasswordEncoder() {
        return new BCryptPasswordEncoder();
    }

    @Override
    protected void configure(HttpSecurity http) throws Exception
{
```

```
        http.csrf().disable()
                .authorizeRequests()
                .antMatchers("/registration",
"/user/add").not().fullyAuthenticated()
                .antMatchers("/audiobooks", "/home",
"/audiobook/{id}").permitAll()
                .antMatchers("/js/**", "/css/**", "/img/**",
"/download/**").permitAll()
                .antMatchers("/admin/**").hasAnyRole("ADMIN",
"SUPER_ADMIN")
                .antMatchers("/user/**").hasAnyRole("USER",
"ADMIN", "SUPER_ADMIN")
                .anyRequest().authenticated()
                .and()
                .formLogin()
                .loginPage("/login")
                .defaultSuccessUrl("/home", true)
                .permitAll()
                .and()
                .logout()
                .logoutSuccessUrl("/home")
                .deleteCookies("JSESSIONID")
                .and()

.exceptionHandling().accessDeniedHandler(accessDeniedHandler);
    }

    @Autowired
    protected void configureGlobal(AuthenticationManagerBuilder
auth) throws Exception {

auth.userDetailsService(userService).passwordEncoder(bCryptPassw
ordEncoder());
    }
}
```

AudiobookFileServiceImpl.java

```
package org.atsynthesizer.demo.service.implementation;


import org.apache.commons.lang3.RandomStringUtils;
import org.atsynthesizer.demo.entity.AudiobookFile;
import org.atsynthesizer.demo.entity.Creator;
import
org.atsynthesizer.demo.repository.AudiobookFileRepository;
import org.atsynthesizer.demo.service.AudiobookFileService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Value;
import
org.springframework.security.core.userdetails.UserDetails;
```

```
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import org.springframework.web.multipart.MultipartFile;

import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.text.DecimalFormat;
import java.text.NumberFormat;
import java.util.Locale;
import java.util.Optional;

@Service
@Transactional
public class AudiobookFileServiceImpl implements
AudiobookFileService {

    @Value("${upload.path}")
    private String UPLOAD_DIR;

    @Autowired
    private AudiobookFileRepository audiobookFileRepository;

    public String getFileSize(File file) {
        long fileSize = file.length();
        NumberFormat formatter = NumberFormat.getInstance(new
Locale("en"));
        formatter.setMaximumFractionDigits(2);
        if (fileSize / (1024 * 1024 * 1024) > 1) {
            return "" + formatter.format((double) fileSize /
(1024 * 1024 * 1024)) + " gb";
        } else if (fileSize / (1024 * 1024) > 1) {
            return "" + formatter.format((double) fileSize /
(1024 * 1024)) + " mb";
        } else if (fileSize / (1024) > 1) {
            return "" + formatter.format((double) fileSize /
(1024)) + " kb";
        } else {
            return "" + formatter.format(fileSize) + " bytes";
        }
    }

    // Save Files
    public String saveUploadedFile(MultipartFile file,
UserDetails currentUser) throws IOException {

        // Make sure directory exists!
        File uploadDir = new File(UPLOAD_DIR +
currentUser.getUsername());
        if (!uploadDir.exists()) {
```

```
            uploadDir.mkdirs();
        }
        if (!file.isEmpty()) {
            String uploadFilePath = currentUser.getUsername() +
"/"
                    + RandomStringUtils.randomAlphanumeric(8) +
"-" + file.getOriginalFilename();

            byte[] bytes = file.getBytes();
            Path path = Paths.get(UPLOAD_DIR + uploadFilePath);
            Files.write(path, bytes);
            return uploadFilePath;
        } else {
            return "";
        }

    }

    @Override
    public AudiobookFile add(AudiobookFile audiobookFile) {
        Optional<AudiobookFile> audiobookFileFromDB =
audiobookFileRepository.findByFilePath(audiobookFile.getFilePath
());
        return audiobookFileFromDB.orElseGet(() ->
audiobookFileRepository.save(audiobookFile));
    }

    @Override
    @Transactional
    public Iterable<AudiobookFile> allAudiobookFiles() {
        return audiobookFileRepository.findAll();
    }

    @Override
    @Transactional
    public AudiobookFile getById(Long id) {
        return audiobookFileRepository.findById(id).get();
    }

    // Save Files
    public String saveBookFile(MultipartFile file, UserDetails
currentUser) throws IOException {

        // Make sure directory exists!
        File uploadDir = new File(UPLOAD_DIR +
currentUser.getUsername());
        if (!uploadDir.exists()) {
            uploadDir.mkdirs();
        }
        if (!file.isEmpty()) {
            String uploadFilePath = currentUser.getUsername() +
"/"
```

```
                    + RandomStringUtils.randomAlphanumeric(8) +
"-" + file.getOriginalFilename();

            byte[] bytes = file.getBytes();
            Path path = Paths.get(UPLOAD_DIR + uploadFilePath);
            Files.write(path, bytes);
            return uploadFilePath;
        } else {
            return "";
        }

    }

}
```

Обозначение	*Наименование*	*Дополнительные сведения*
	Текстовые документы	
БГУИР ДП 1-40 01 01 01 029 ПЗ	Пояснительная записка	107 с.
	Отзыв руководителя	
	Рецензия	
	Графические документы	
ГУИР.751003-01 СП	Приложение для синтеза,	Формат А1
	хранения и распространения	
	аудиокниг	
	Схема программы	
ГУИР.751003-01 СА	Движение данных при	Формат А1
	регистрации	
	Схема алгоритма	
ГУИР.751003-02 СА	Контроллер аудиокниг	Формат А1
	Схема алгоритма	
ГУИР.751003-01 ПЛ	Диаграмма развёртывания	Формат А1
	приложения	
	Плакат	
ГУИР.751003-02 ПЛ	Диаграмма UML приложения	Формат А1
	Плакат	
ГУИР.751003-03 ПЛ	База данных приложения	Формат А1
	Плакат	

					БГУИР ДП 1-40 01 01 01 029 Д1
Изм	Лист	№ докум.	Подп.	Дата	
Разработал		Гринчик В.В.			Веб-приложение для синтеза, хранения и распространения аудиокниг на базе Spring Framework. Ведомость дипломного проекта
Проверил		Можей Н.П.			
Т.контроль		Можей Н.П.			
Н.контр.		Манцевич А.В.			
Утвердил					

Литера	Лист	Листов
т	107	107

Кафедра ПОИТ
гр. 751003