Санкт-Петербургский государственный университет

Программная инженерия

Кудряшова Анна Александровна

Облачная платформа для хранения и редактирования 3D-моделей человека

Бакалаврская работа

Научный руководитель:
к.т.н. доцент Литвинов Ю. В.

Рецензент:
Пенкрат Н. А.

Санкт-Петербург
2018

SAINT PETERSBURG STATE UNIVERSITY

Software engineering

Anna Kudriashova

Cloud platform for storing and viewing 3D models of a human body

Graduation Thesis

Scientific supervisor:
assistant professor Yurii Litvinov

Reviewer:
Nikolay Pencrat

Saint Petersburg
2018

Оглавление

Введение

В последние годы в мире набирает популярность здоровый образ жизни. По этой причине люди все чаще задумываются о состоянии своего здоровья и следят за ним. Как известно, в результате обследования человек получает документы или изображения, демонстрирующие результат данной процедуры. В результате, после прохождения нескольких обследований скапливается множество документов. Перевозить их не всегда удобно, так как они имеют большой формат, что может повлечь за собой их деформацию, вследствие чего информация, хранящаяся на них, может быть частично или полностью повреждена. Также со временем снимки имеют свойство выцветать и после этого информацию на них трудно идентифицировать.

Помимо этого, в настоящее время более широкое распространение получает 3D-сканирование [1]. 3D-моделирование представляет собой процесс создания трёхмерной модели объекта. Данный процесс заключается в построении геометрической проекции трёхмерной модели на плоскость с помощью специальных программ. Получило развитие коммерческое 3D-сканирование, появляются компании, которые предлагают свои услуги по проведению 3D-сканирования, в результате, любой желающий может получить объемную модель своего тела. Данные модели широко применяются в медицине, в частности в пластической хирургии. Их использование позволяет визуализировать будущие изменения внешности, что значительно упрощает взаимодействие между доктором и пациентом.

Сервис для хранения и просмотра снимков и 3D-моделей необходим как работникам медицинской сферы, так и пациентам. Возможность хранить все медицинские снимки и данные в одном месте позволяет оптимизировать и ускорить процесс обмена информацией между доктором и пациентом, что положительно сказывается на их взаимодействии. Не менее важным фактором является экономия времени, человек не тратит его, чтобы найти нужные снимки, они все хранятся в одном месте. При создании приложения большую роль играет доступность, с

этой задачей справляются веб-приложения, так как они не зависят ни от типа устройства, ни от его платформы, для них важен только доступ к сети интернет.

В данной дипломной работе была поставлена задача разработать веб-сервис, который отвечает данным требованиям.

1. Постановка задачи

Целью данной работы является создание веб-сервиса, позволяющего сохранять и просматривать 3D-модели и снимки тела человека. Для достижения поставленной цели нужно было выполнить следующие задачи:

- проанализировать существующие решения в области хранения изображений и 3D-моделей;
- разработать архитектуру приложения;
- реализовать прототип, обладающий следующей функциональностью:
    - верификация аккаунта пользователя;
    - хранение 3D-моделей и снимков в облаке;
    - демонстрация 3D-модели;
    - добавление и удаление 3D-модели и снимков;
- разработать интерфейс приложения.

2. Обзор предметной области

2.1. Обзор существующих решений

В ходе работы были проанализированы существующие решения в области хранения и просмотра 3D-моделей и изображений. Приложения оценивались по следующим критериям:

- возможность загружать и скачивать 3D-модели и изображения;
- возможность просматривать 3D-модели и изображения;
- возможность использовать приложение в браузере.

В результате были изучены следующие сервисы:

- Naked3D[11] — сервис позволяющий визуализировать прогресс, достигнутый в результате тренировок. Для отслеживания результата с помощью портативного домашнего сканера создается 3D-модель тела человека. Сервис предоставляет доступ к мобильному приложению и позволяет просматривать 3D-модели полученные с портативного сканера;
- emb3d[12] — приложение для просмотра и хранения 3D-моделей на мобильных устройствах. Данное приложение позволяет рассматривать модели под различными углами;
- Google Photo[18] — сервис для хранения изображений. Данный сервис позволяет загружать, просматривать и скачивать различные виды изображений.

После анализа существующих решений было установлено, что все эти сервисы не позволяют загружать и просматривать как 2D-изображения, так и 3D-модели.

2.2. Используемые инструменты и технологии

При разработке приложения применялись следующие технологии:

- основным языком выступает Java[3]. Одним из преимуществ данного языка является его широкая распространенность, что обеспечивает наличие фреймворков и библиотек для различных целей;
- для работы с базой данных используется MySQL[5]. MySQL — самая популярная реляционная база данных с открытым исходным кодом. Благодаря своей надежности и простоте чаще всего применяется при разработке web-приложений;
- для облегчения взаимодействия с базой данных используется библиотека Hibernate[6], предназначенная для решения задач объектно-реляционного отображения. Благодаря Hibernate нет необходимости вручную писать SQL-код, он генерируется автоматически;
- для обеспечения безопасности в процессе регистрации и авторизации пользователя используется фреймворк Spring[17], в частности Spring Boot и Spring Security. Данный фреймворк позволяет осуществлять валидацию email при регистрации и обеспечивает взаимодействие с базой данных;
- в качестве контейнера сервлетов выступает Apache Tomcat[7], так как является встроенным в фреймворк Spring, что позволяет с легкостью настроить сервер;
- при разработке фронтенда используется фреймворк Bootstrap[16] и движок шаблонов Thymeleaf[10]. Bootstrap включает в себя HTML и CSS-шаблоны для создания эргономичного и эстетичного web-интерфейса;
- в качестве облачного хранилища был выбран сервис Amazon S3[13]. Хранилища Amazon предоставляют как возможность бесплатного ограниченного использования, которое может применяться при тестировании, так и коммерческие варианты;

- для демонстрации 3D-моделей используется библиотека Three.js[2]. Данная библиотека поддерживает работу с большинством современных форматов, используемых при 3D-моделировании. Помимо этого данная библиотека написана на JavaScript[4], что удобно при написании веб-сервисов.

3. Архитектура

При создании сервиса была разработана архитектура, представленная на рисунке 1.

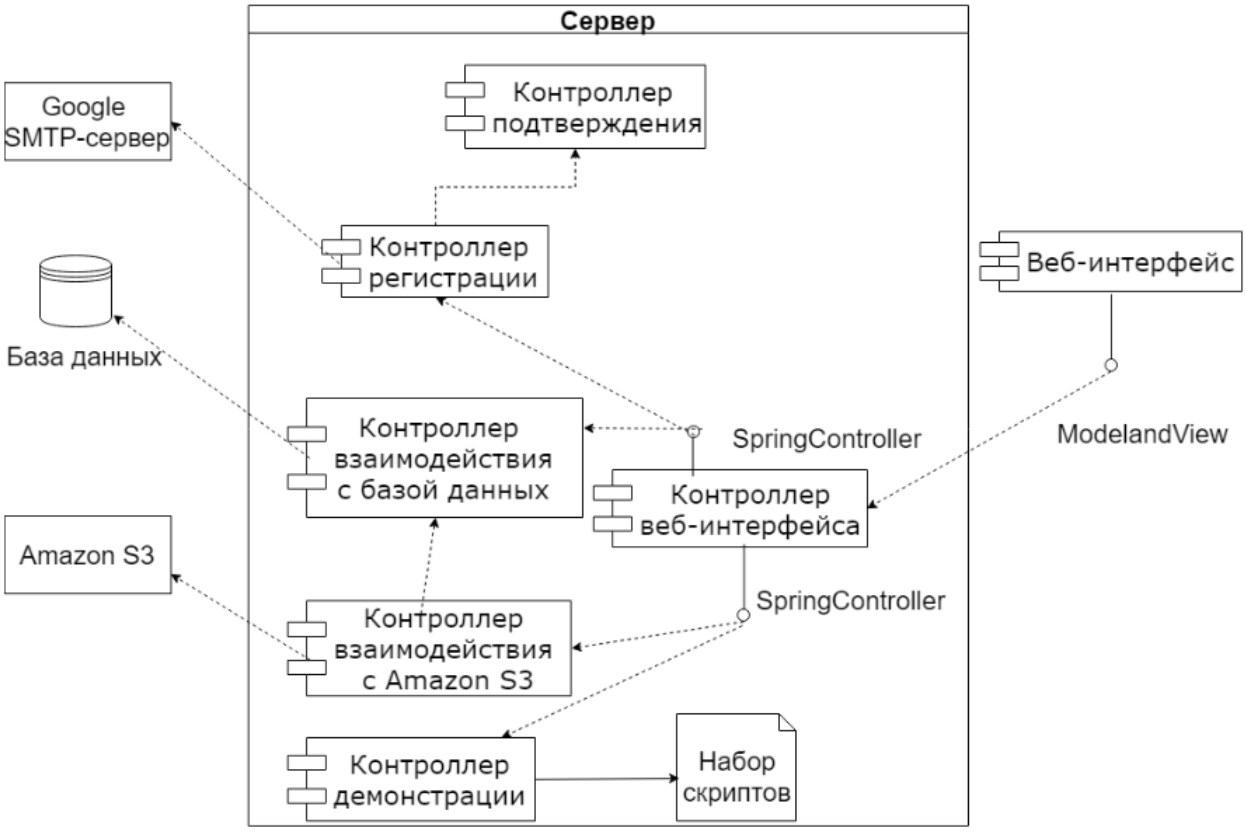

Рис. 1: Диаграмма компонент архитектуры веб-сервиса

Взаимодействие между различными компонентами сервиса осуществляется посредством контроллеров. Spring Framework предоставляет REST API для взаимодействия с различными частями системы: облачным хранилищем, базой данных, сервером отправки сообщений и основным сервером. Каждая функция сервиса управляется своим собственным контроллером.

Взаимодействие между пользователем и основным сервером осуществляется через веб-интерфейс. Сервер обеспечивает сохранение и обновление информации в базе данных, также регулирует процесс регистрации и валидации аккаунта. Помимо этого, сервер обеспечивает сохранение и скачивание пользовательских данных из облачного хранилища Amazon S3. Для хранения 3D-модели используются нескольких

файлов: описание точек модели и файл текстуры. Контроллер демонстрации осуществляет выбор скрипта для отображения необходимых данных, так как модель может иметь различный формат и способ представления текстуры, что влияет на способ загрузки и демонстрации.

4. Реализация

4.1. База данных

Информация о пользователе содержится в таблице User, а информация о файле хранится в таблице Upload_files. Для предоставления пользователю возможности обладать несколькими файлами, отношение многие ко многим реализовано через вспомогательную таблицу User_file. 3D-модель может состоять из нескольких файлов: файла точек модели и файла текстуры, а может быть представлена как набор точек без файла текстуры, поэтому для корректного отображения модели в базу данных сохраняется расширение и url-адрес файла текстуры. Таблица Role предназначена для хранения ролей пользователей, которые обеспечивают различный уровень привилегий, что позволяет предоставлять пользователям различный уровень доступа к сервису. Связь с таблицей User осуществляется через таблицу User_role. Схема базы данных представлена на рисунке 2.

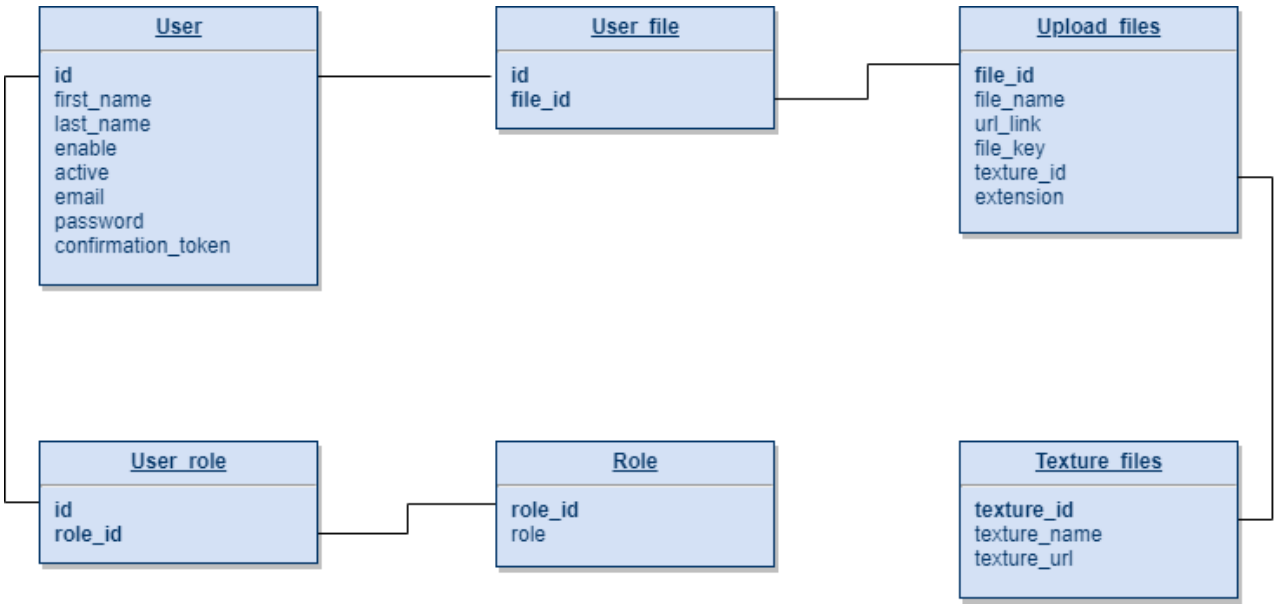

Рис. 2: Схема базы данных веб-сервиса

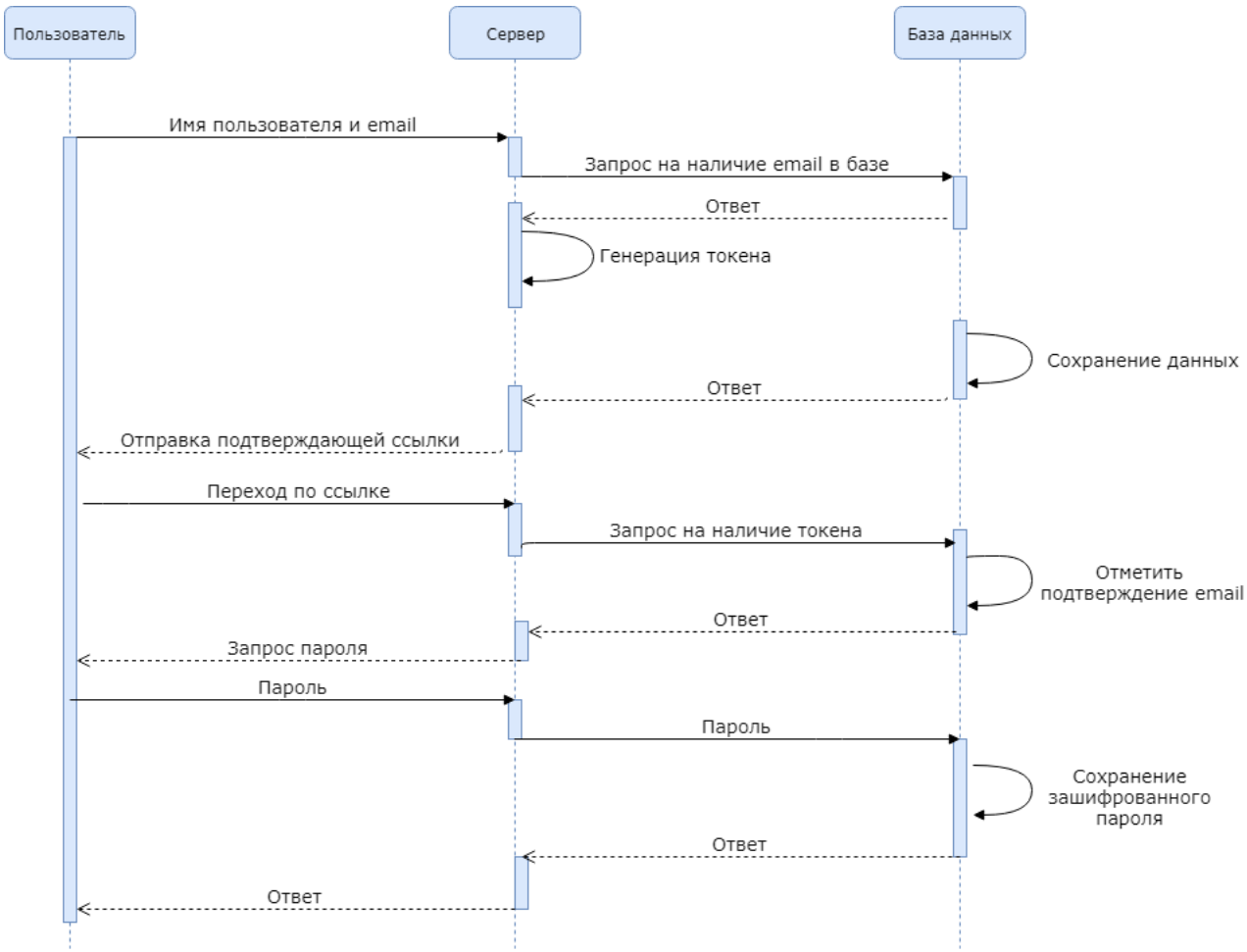

Рис. 3: Диаграмма последовательностей процесса регистрации пользователя

4.2. Регистрация пользователя

Процесс регистрации состоит из двух этапов. Первый включает в себя указание имени, фамилии и адреса электронной почты, эти данные сохраняются в базу данных, после чего с помощью Spring Security создается подтверждающий токен, который также сохраняется в базу. Данный этап необходим, чтобы обеспечить уникальность и истинность подтверждающей ссылки, которая будет отправлена на указанный адрес электронной почты. Структура подтверждающей ссылки выглядит следующим образом: *адрес сервера/confirm?token=confirmation_token*. После перехода по ссылке пользователь помечается в базе данных, как прошедший валидацию и ему предоставляется возможность установить пароль для своего аккаунта. С помощью библиотеки Zxcvbn проверяет-

ся сложность и распространенность пароля. Зарегистрированные пользователи попадают в группу USER, с ограничением доступа к некоторым страницам. Диаграмма последовательностей для процесса регистрации представлена на рисунке 3.

4.3. Аутентификация

Процесс аутентификации обеспечивается с помощью фреймворка Spring Security. На странице входа пользователю необходимо ввести адрес электронной почты и соответствующий пароль, после осуществляется проверка наличия данного пользователя в базе данных. В случае положительного ответа проверяется корректность пароля, после успешного прохождения данного этапа пользователь помечается в базе данных, как активный, это необходимо для обеспечения возможности выйти из аккаунта в случае необходимости.

4.4. Загрузка и скачивание модели

В большинстве случаев 3D-модель состоит из нескольких файлов: файла текстуры и файла описания точек. По этой причине необходимо обеспечить связь между различными файлами для скачивания и демонстрации моделей.

При загрузке 3D-модели пользователь может выбрать несколько файлов одновременно, но нельзя гарантировать в каком порядке эти файлы будут загружены. Для решения данной проблемы файлы текстур загружаются в отдельный TextureRepository — репозиторий, в котором хранятся все загруженные текстуры. После загрузки файла точек модели осуществляется поиск необходимой текстуры в репозитории, необходимым условием является совпадение имени файла точек и файла текстуры. В случае, если у модели отсутствует файл текстуры, ей будет присвоена стандартная. Стандартная текстура была загружена в облако и назначается всем моделям, текстура которых не определена.

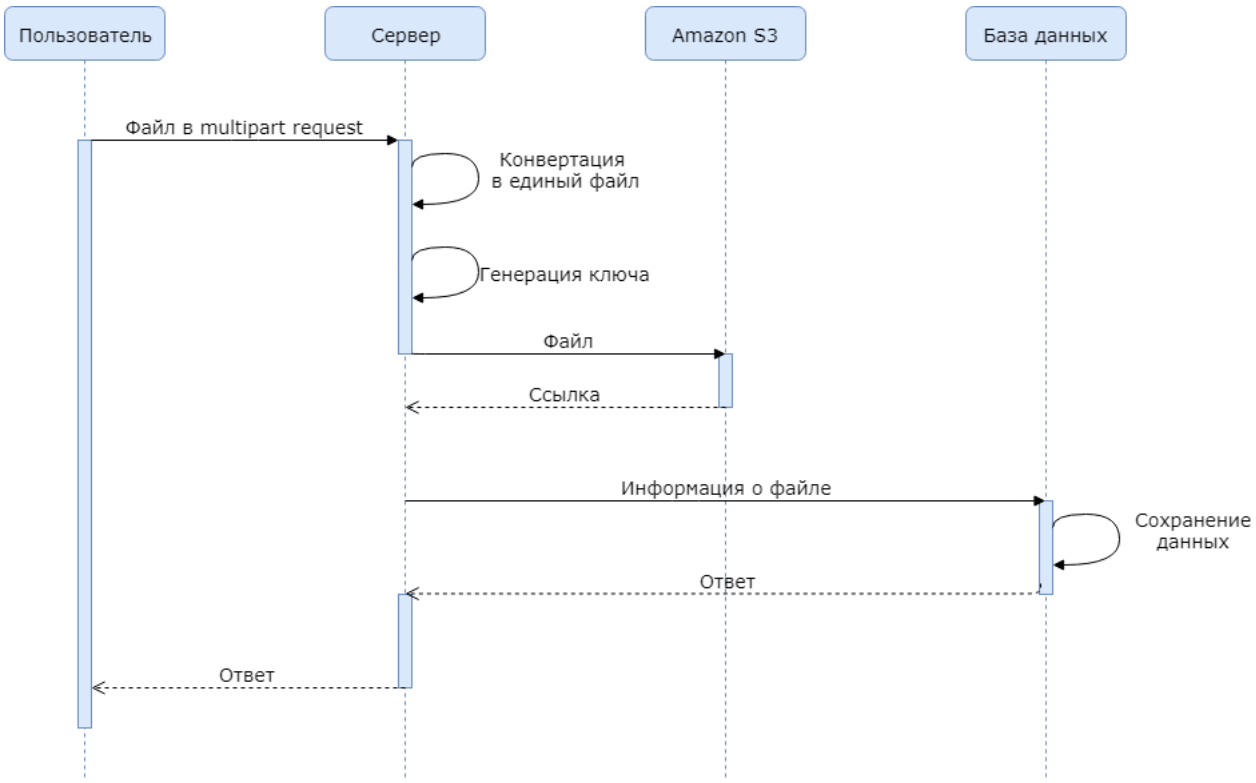

Рис. 4: Диаграмма последовательности процесса загрузки файла в облачное хранилище Amazon S3

Загрузка файла в облако осуществляется в несколько этапов. На первом этапе происходит выбор файла и загрузка его во временое хранилище с помощью multipart request, далее происходит преобразование файла из MultipartFile в File. По окончанию пользовательской сессии временное хранилище очищается. На втором этапе осуществляется отправка файла в облако, с помощью метода saveToAmazonS3. Метод написан с помощью фреймворка Spring, который предоставляет API для взаимодействия с облачным хранилищем. После успешной загрузки информация о файле сохраняется в базу данных, включая url-адрес файла для изображения или url-адреса файла точек и текстуры для 3D-модели, которые выглядят следующим образом:

endpointUrl/bucketName/key, где

- *endpointUrl* — сервер Amazons S3, который зависит от выбранного региона;
- *bucketName* — имя репозитория, который связан с сервисом;

- *key* — уникальное имя, присваемое каждому файлу, которое состоит из текущего времени в секундах и изначального имени файла.

Диаграмма последовательности процесса загрузки представлена на рисунке 4.

Скачивание модели или изображения осуществляется с помощью API для взаимодействия с Amazon S3 по ключу объекта, который хранится в базе данных. Если модель состоит из нескольких файлов, то они скачиваются последовательно. После скачивания объект преобразуется в новый файл с расширением, которое было указано в базе.

4.5. Демонстрация модели

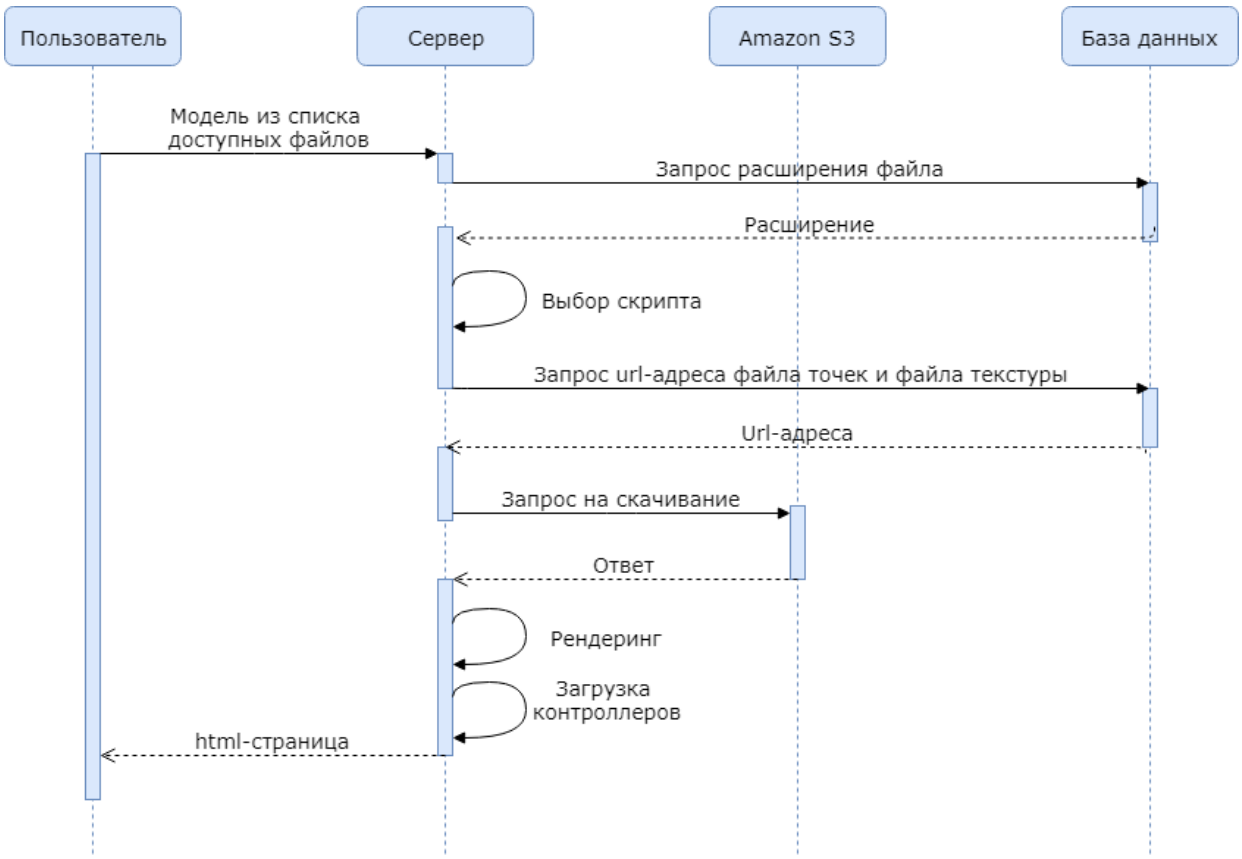

Рис. 5: Диаграмма последовательности демонстрации 3D-модели

3D-модель может быть представлена в различных форматах: .FBX, .3DS, .dxf, .obj и другие. В данной работе поддерживается демонстрация моделей в формате .obj, другие форматы можно добавить, написав

соответствующий скрипт. Унификация сервиса с помощью преобразования модели к формату JSON оказалась нецелесообразной, так как для осуществления такого действия требует обращения к сторонним приложениям или сервисам, например Clara.io[9] или Blender[8], которые не имеют API для взаимодействия с фреймворком Spring. По этой причине было принято решение использовать свой скрипт для каждого формата модели.

Демонстрация модели реализована на языке JavaScript с помощью библиотеки Three.js. При выборе модели из списка доступных пользователю файлов необходимый скрипт для демонстрации выбирается на основе расширения выбранного файла. Url-адреса файла точек и текстуры извлекаются из базы данных и передаются в качестве аргументов в адресе html-запроса. В случае, если аргумент f_texture равен пустой строке, то считается, что файл текстуры отсутствует.

Модель представляет собой файл с описанием точек и файл текстуры. В свою очередь текстура может быть представлена несколькими способами:

- как файл в формате .mtl, где содержится описание нескольких .jpg или .png файлов, используемых для хранения текстуры;
- как один файл в формате .png или .jpg, если текстура модели не многослойна;
- текстура модели может отсутствовать.

После загрузки модели осуществляется рендеринг сцены и объекта. В среднем рендеринг занимает 100 миллисекунд. Пользователь может приближать и отдалять модель, также он имеет возможность поворачивать модель вдоль любой из осей под необходимым углом. Вращение осуществляется с помощью метода OrbitControls, что позволяет достичь плавного поворота 3D-модели. Диаграмма последовательности процесса демонстрации 3D-модели представлена на рисунке 5.

5. Веб-сервис

В ходе написания дипломной работы был разработан прототип веб-сервиса, обладающий следующей функциональностью:

- регистрация с верификацией аккаунта;
- аутентификация;
- выход из аккаунта;
- загрузка 3D-модели;
- загрузка изображения;
- скачивание изображения;
- демонстрация 3D-модели с управлением камерой;
- просмотр изображения.

5.1. Сценарий использования

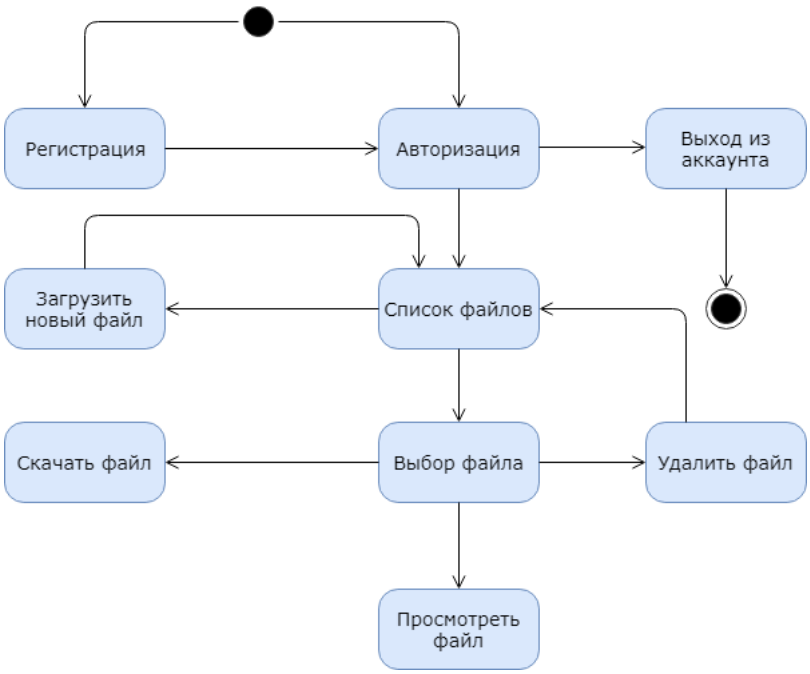

Рис. 6: Диаграмма состояний с точки зрения пользователя

Для взаимодействия с сервисом пользователю необходимо зарегистрироваться или пройти авторизацию. После осуществления данного действия предоставляется возможность просмотреть список файлов, скачать файл или загрузить его. При просмотре 3D-модели пользователь имеет возможность приближать или отдалять модель, а также поворачивать её вдоль любой оси. Диаграмма состояний с точки зрения пользователя представлена на рисунке 6.

5.2. Интерфейс

Для обеспечения взаимодействия пользователя и веб-сервиса был разработан прототип графического интерфейса. Структурно интерфейс разделён на несколько областей: навигационная панель, панель действий и основной контейнер, в зависимости от текущей страницы его наполнение различно.

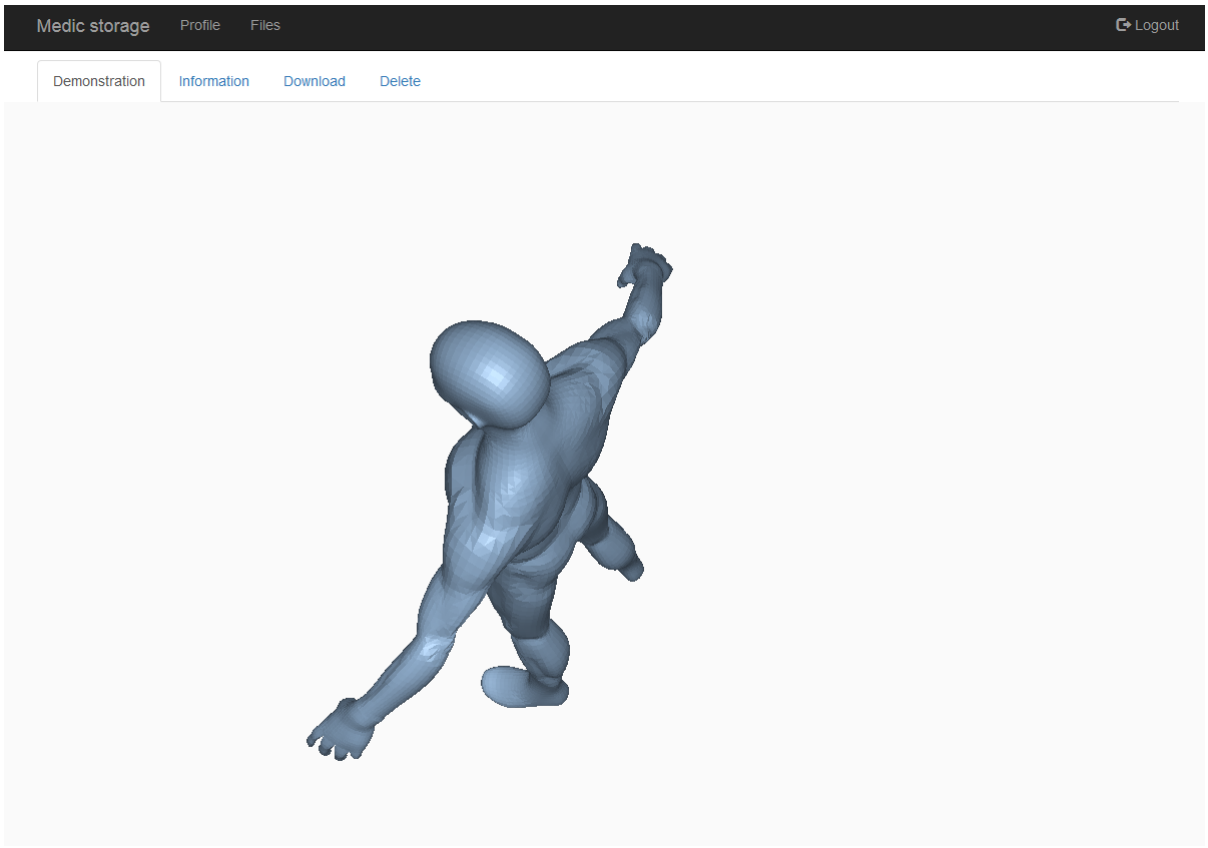

Рис. 7: Демонстрация 3D-модели со стандартной текстурой

При создании интерфейса использовался фреймворк Bootstrap. Данный фреймворк позволяет создать простой, эргономичный интерфейс, который не перегружает рецепторы визуального восприятия пользователя. Взаимодействие между функциональной частью и интерфейсом обеспечивается фреймворком Spring и движком шаблонов Thymeleaf с использованием интерфейса ModelandView.

На рисунках 7 и 8 представлен интерфейс приложения. Также здесь продемонстрирована возможность отображения различных типов 3D-моделей с возможностью управления камерой.

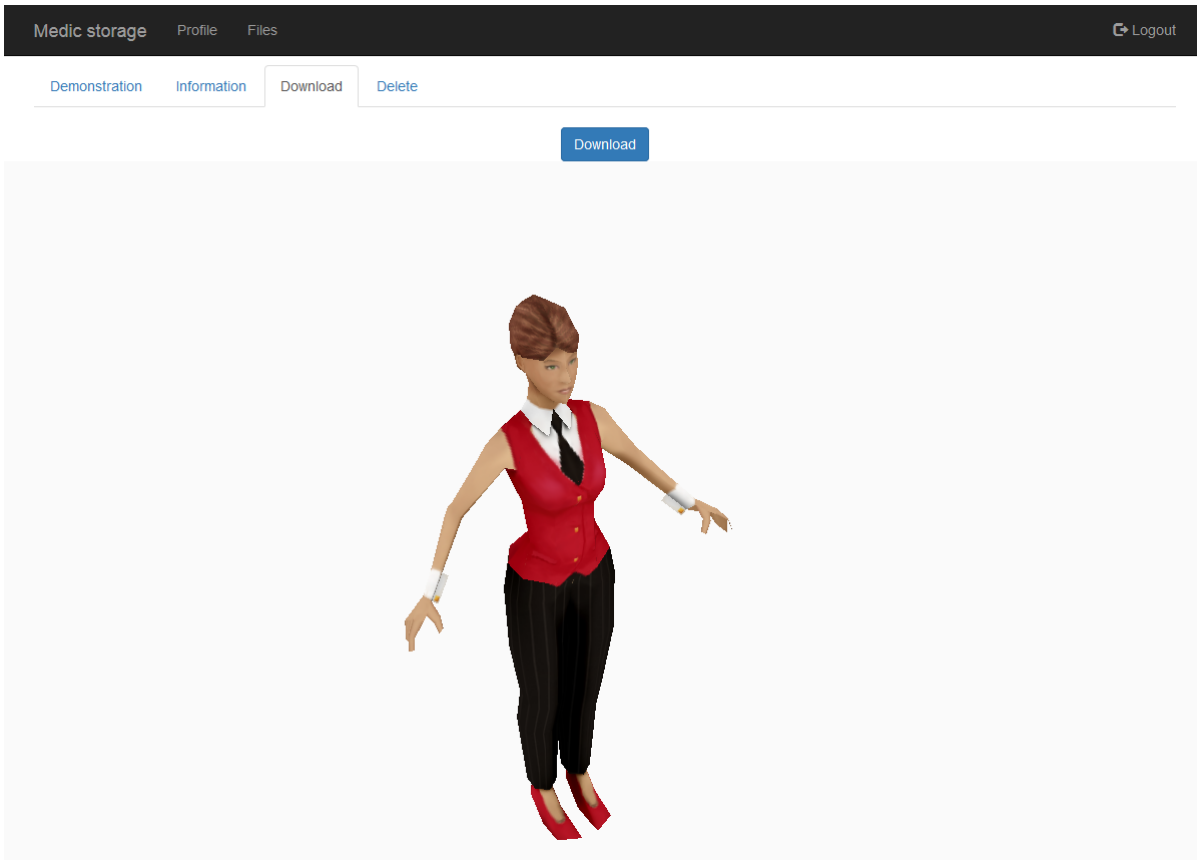

Рис. 8: Демонстрация 3D-модели с многослойной текстурой

6. Тестирование

Разработанный в рамках данной работы веб-сервис был запущен локально. Ключевые функции были протестированы с использованием 3D-моделей с различным количеством полигонов. Модели, используемые при тестировании были загружены с бесплатных разделов стоковых хранилищ Turbosquid[14] и Cgtrader[15]. Для тестирования применялось оборудование, обладающее следующими характеристиками: Intel Core i5-3210M CPU 2.50 GHz RAM 4 GB. Результаты тестирования представлены в таблице 1. В частности было отмечено, что для высокополигональных моделей время рендеринга составляло 5056 миллисекунд, что является неудовлетворительным результатом. Для решения данной проблемы в дальнейшем предлагается использовать алгоритмы сжатия модели, что поможет сократить время рендеринга.

Таблица 1: Время рендеринга 3D-моделей с различной полигональностью

Модель	**Кол-во полигонов**	**Первый рендеринг (ms)**	**Ср. рендеринг 30 изм.(ms)**
Femal_Base_Mesh	13,892	218	198
male_head	3,242	268	107
headforprint	1,000,000	5056	4590

Результаты

В ходе выполнения выпускной квалификационной работы были достигнуты следующие результаты:

- Проведен анализ существующих решений в области хранения и демонстрации 3D-моделей и изображений;
- Разработана архитектура приложения;
- Разработан прототип облачной платформы для хранения и демонстрации 3D-моделей и изображений;

  Разработан прототип облачной платформы, предоставляющая возможность регистрации, двухфакторной аутентификации, загрузки, удаления и демонстрации 3D-моделей и изображений.
- Разработан графический интерфейс, обеспечивающее взаимодействие пользователя с платформой.

Список литературы

[1] Scott Telfer James Woodburn. The use of 3D surface scanning for the measurement and assessment of the human foot // Journal of Foot and Ankle Research. — 2010. — URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2944246/pdf/1757-1146-3-19.pdf (online; accessed: 20.04.2018).

[2] Библиотека Three.js. — URL: https://github.com/mrdoob/three.js/ (online; accessed: 23.05.2018).

[3] Документация Java SE версии 1.8. — URL: https://docs.oracle.com/javase/8 (online; accessed: 20.04.2018).

[4] Документация JavaScript. — URL: https://www.javascript.com/ (online; accessed: 23.05.2018).

[5] Документация MySQL Server. — URL: https://dev.mysql.com/doc/refman/8.0/en/ (online; accessed: 20.04.2018).

[6] Документация библиотеки Hibernate. — URL: http://hibernate.org/orm (online; accessed: 20.04.2018).

[7] Сайт Apache Tomcat. — URL: http://tomcat.apache.org (online; accessed: 20.04.2018).

[8] Сайт Blender. — URL: https://www.blender.org/ (online; accessed: 23.05.2018).

[9] Сайт Clara.io. — URL: https://clara.io/ (online; accessed: 23.05.2018).

[10] Сайт движка шаблонов Thymeleaf. — URL: https://www.thymeleaf.org (online; accessed: 20.04.2018).

[11] Сайт приложения Naked3D. — URL: https://naked.fit (online; accessed: 20.04.2018).

[12] Сайт приложения emb3D. — URL: https://www.emb3d.com (online; accessed: 20.04.2018).

[13] Сайт сервиса Amazon S3. — URL: https://aws.amazon.com/s3 (online; accessed: 20.04.2018).

[14] Сайт стокового хранилища Turbosquid. — URL: https://www.turbosquid.com/Search/3D-Models/free (online; accessed: 28.05.2018).

[15] Сайт стокового хранилища Cgtrader. — URL: https://www.cgtrader.com/free-3d-models (online; accessed: 28.05.2018).

[16] Сайт фреймворка Bootstrap. — URL: https://getbootstrap.com (online; accessed: 20.04.2018).

[17] Сайт фреймворка Spring. — URL: https://spring.io (online; accessed: 20.04.2018).

[18] Сервис Google Photo. — URL: https://photos.google.com (online; accessed: 20.04.2018).