Здравствуйте, уважаемая комиссия. В качестве введения, я бы хотел рассказать вам о том, что такое DAM система. DAM или же digital asset management система, на русский язык можно перевести как система управления электронными документами. Данная система пользуется большой популярностью в больших организациях, где сотрудникам приходится работать с большим потоком медиа контента, такого как изображения, видео, аудио. Данные системы могут хранить так же и обычные документы, но это происходит не часто. Данные файлы приходится хранить где-то и обрабатывать, но очень часто нет единых правил для работы с этими документами. Нет общего хранилища, т.е. у каждого сотрудника свой набор файлов, в своих папках и т.д. Это удобно, когда с файлами работает один сотрудник, но, когда с ними должны работать и другие сотрудники – появляются проблемы. Отсутствие единого хранилища файлов усложняет обмен ими между сотрудниками. Бывают файлы, которые имеют размер более 500 мб и такие файлы очень долго и не удобно отправлять кому-либо. Отсутствие версионности файлов не позволяет быть всегда в курсе последних изменений, т.е. результатов работы сотрудника никто не увидит, пока он не вышлет новые копии всем заинтересованным лицам. Это отнимает очень много времени и сил сотрудников. Для того что бы решить данные проблемы и используются DAM системы. Они предоставляют единое, защищенное хранилище файлов, которое обладает множеством функций. Пользователи имеют возможность зайти на сайт и найти все файлы, которые им нужны за пору кликов. Причем сделать это можно с разных устройств. Обмен данными также происходит очень быстро, пользователю необходимо всего лишь отправить ссылку на файл. DAM система поддерживает версионность и пользователи увидят что добавилась новая версия файла, как только другой пользователь ее загрузит. Это происходит моментально и без каких-либо сложностей. Но это далеко не все возможности DAM систем. Еще одна возможность заключается в том, что файлы можно классифицировать по их типу, назначению и т.д. Классификации можно добавлять произвольное количество и из них можно строить дерево, как в структуре папок. Но главным отличием классификаций от папок является то, что классификации хранятся просто как список значений. Т.е. если говорить про структуру папок, что бы добавить 1 файл в две разные классификации (папки) нужно было бы скопировать данный файлы в эти папки. А что, если классификаций должно быть больше 2-х? копировать файл размером более 500 мб. Очень расточительно по отношению к дисковому пространству. DAM система же не хранит файлы в структуре папок, поэтому добавление классификации файлу является очень дешевой операцией. После того, как файлы классифицированы, их можно легко найти, отфильтровав по классификациям.

Так же очень часто необходимо хранить какие –то вспомогательные данные о файле. Понятное дело, что файлы несут в себе какую-то информацию. В файле картинки хранятся данные о размерах, в видео о продолжительности, но этого часто не хватает. К примеру, компания занимается производством маек и планирует запустить маркетинговую компанию для рекламы своего нового продукта. Фотографы сделали множество фотографий моделей в новых майках и сохранили их в DAM системе. Добавили классификации «Лето», «Мужская коллекция» к примеру, но что, если необходимо найти все майки, красного цвета или же определённого размера. Создавать всякий раз новые классификации для цветов или размеров не очень удобно. Поэтому существует такое понятие как «поля» ассета. Так называемые метаданные продукта. Поля появляются на ассете когда его относят к классификации. Те поля, которые привязаны к классификациям на ассете появятся и их можно будет заполнить любой необходимой информацией. Допустим, поле цвет, его можно заполнять значениями и это не будет проблемой, в самой системе не нужно будет ничего нового создавать. Далее по значениям полей можно осуществлять поиск и это очень сильно упрощает работу с файлами.

Такие системы идут в комплекте с программными библиотеками, которые можно использовать в своих модулях и эти библиотеки обеспечивают программиста всеми возможностями для работы с системой. Это дает возможность подстраивать систему под бизнес требования.

**Функции, которые реализованные в данном дипломном проекте представлены на первом плакате (use case).**

В данном дипломном проекте, я попытался реализовать упрощенную версию данной системы. Поддерживаются основные функции как загрузка файла, добавление классификаций, добавление полей, присваивание значений полям. Поиск получился очень упрощенным, так как обработка поисковых запросов очень сложная задача. Поэтому искать файлы можно по классифиации, значению поля и идентификатору. Настоящие системы поддерживают возможность фильтрации по множеству полей и классификаций, но парсинг запроса, построение логических выражений довольно трудная задача, особенно когда данные хранятся в нескольких базах разного типа.

Как я уже сказал, было использовано две базы данных: PostgreSQL и elasticsearch. Данные ассета хранятся в базе elasticsearch, и часть из них храниться в PostgreSQL базе. База данных PostgreSQL больше расчитана на хранение информации по полям, классификациям, пользователям и т.д. Потому что необходимо поддерживать связи между данными сущностями и строить запросы используя данные связи для быстрого поиска нужной информации. База данных elasticsearch, в отличии от PostgreSQL является не реляционной. Данные в ней хранятся как текст формата JSON и данная база не поддерживает связи внутри одного документа. Данная база используется потому, что она обеспечивает очень быстрый поиск данных и это является ключевым фактором. При загрузке ассета, отображается та информация, которая храниться в elasticsearch базе, но если пользователю необходимо получить больше информации – происходит отправка запроса в базу PostgreSQL. Это так же удобно и при автоматизации процессов. Данные грузятся по мере надобности что позволяет увеличить производительность всей системы целиком.

Для разработки ПС был выбран язык программирования Rust. Это системный язык, уровня C/C++ в котором нет сборщика мусора, но и нет ручного управления памятью. Он компилируется в нативный код при помощи LLVM и производительность сравнима с C/C++. Так же данный язык очень безопасен в отличии от тех же C/C++ и даже более высокоуровневых языков со сборщиками мусора, таких как C#, Java и т.д. Данные качества, как скорость и являются очень важными, при разработке ПС, которые должны справляться с высокой нагрузкой и иметь малую вероятность сбоя в работе.

**На диаграмме классов можно увидеть архитектуру ПС**

Параметрический полиморфизм; мономорфизация; ad hoc полиморфизм

**На блок схемах вы можете увидеть схему работы ПС, алгоритм регистрации и алгоритм редактирования ассета.**

**На последнем плакате представлены скриншоты программного средства.** На первом скриншоте изображена главная страница ПС. Справа можно увидеть дерево классификаций и отображение ассетов конкретной классификации.

**На втором скриншоте** можно увидеть, как выглядит подробная информация об ассете т.е. список полей и их значений.