Логирование

Необходимо сделать логирование действий пользователя и действий системы. Данные должны быть подвержены анализу со стороны человека, но не машины (вывод для автопарсинга не нужен). Последние данные действий пользователя должны находится в быстром доступе для человека.

# Общие положения логирования

Логирование должно вестись в текстовые файлы и в БД postgresql. Предусмотреть возможность безболезненного перехода на Elasticsearch (опционально)

Структура папок текстовых файлов следующая: Полный путь до логов\Начальная папка\Год\Месяц\День\Файл.csv

Путь к папке с логами должен быть настраиваемым извне как для пользовательских логов, так и для системных логов

Текстовый файл должен быть в формате “ДАТА\_ЛОГИН\_IP.csv”. Для сервера логин и IP опускаются, но добавляет тип лога(info/error).

Строка лога в csv должна быть следующей структуры, с разделителем “;” :

1. дата-время создания лога (совершения действия системой)
2. IP компьютера пользователя (опускается для лога системы)
3. ID пользователя в системе (опускается для лога системы)
4. тип лога, определяется ниже
5. сообщение лога, определяется ниже

# Логирование действий пользователя

Каждое действие пользователя с документами и поручениями, приводящее к мутации документа или поручения, должно быть залогировано

Каждое сообщение в логах при отображении или выгрузки должно начинаться со следующей структуры: «{ГГГГ-ММ-ДД ЧЧ:ММ:СС}:{IP пользователя}:{Логин пользователя}:{Тип лога}:»

Доступные типы лога и шаблон сообщения:

1. Логин
   1. Совершен логин с компьютера {Имя компьютера в сети}
2. Логаут
   1. Совершен логаут с компьютера {Имя компьютера в сети}
3. Создание документа
   1. Создан документ {ID: Наименование документа}
4. Согласование документа
   1. Документ {ID: Наименование документа} согласован {ID: ФИО согласовавшего пользователя} на этапе {номер этапа согласования} со статусом {статус согласования(согласован/не согласован/согласован с замечанием)}
5. Отклонение документа
   1. Документ {ID: Наименование документа} отклонен с причиной {Причина отклонения}
6. Возврат документа на доработку
   1. Документ {ID: Наименование документа} возвращен создателю на доработку с причиной {Причина возврата}
7. Возврат документа на предыдущий шаг согласования
   1. Документ {ID: Наименование документа} возвращен предыдущему подписанту {ID: ФИО предыдущего подписанта} с причиной {Причина возврата}
8. Создание поручения по документу
   1. По документу {ID: Наименование документа} создано поручение с ID={ID поручения} на пользователя {ID: ФИО исполнителя}
9. Выполнение поручения по документу
   1. По документу {ID: Наименование документа} выполнено поручение с ID={ID поручения}
10. Загрузка файла
    1. По документу {ID: Наименование документа} загружен файл {ID:Наименование документа}
11. Просмотр/скачивание файла
    1. По документу {ID: Наименование документа} просмотрен файл {ID:Наименование документа}
12. (для Михеевой) Подтверждение поручения по документу
    1. По документу {ID документа: Наименование документа} подтверждено поручение с ID={ID поручения}
13. Документ переведен на регистрацию
    1. Документ {ID: Наименование документа} переведен на регистрацию документа в ООПЗ
14. Документ зарегистрирован
    1. Документ {ID: Наименование документа} зарегистрирован в ООПЗ {Дата регистрации ГГГГ-ММ-ДД: Номер регистрации}
15. Документ исполнен
    1. Документ {ID: Наименование документа} переведен в Исполненные
16. Перевод документа в Архив(пользователь нажал кнопку перевода)
    1. Документ {ID документа: Наименование документа} досрочно переведен в Архив

Вечером раз в день в 20.00 происходит выгрузка данных за текущий день в текстовые файлы, формат описан выше. Для каждого пользователя будет отдельный текстовый файл за текущий день.

Путь для хранения текстовых файлов начинается с Начальной папки по имени ClientLogs

После выгрузки данных за последний день происходит проверка БД и удаляются все записи, срок создания которых превышает 31 день от текущего момента, **срок хранения логов в БД должен быть настраиваемым извне**

# Логирование действий системы

Так как нам важно не только какую команду послал пользователя, но и как она выполнилась в системе, нам необходимо вести лог работы системы.

Доступные типы лога и шаблон сообщения:

1. INFO: Запуск системы
   1. Система запущена. Ожидаю клиента по адресу {адрес клиента}. API доступен по адресу {адрес сервера}. Отслеживание задач {Запускать ли scheduler}
2. ERROR: Ошибка запуска системы
   1. Сбой запуска. Ошибка: {Ошибка}, где Ошибка – стандартный вывод Nodejs об ошибке
3. INFO:Предоставление информации клиенту
   1. {Endpoint}:{Параметры, переданные на endpoint}. Подразумеваем, что при создании этого лога результат предоставления информации успешен
4. ERROR:Ошибка предоставления информации клиенту
   1. {Endpoint}:{Параметры, переданные на endpoint}:{Ошибка}, где Ошибка – стандартный вывод Nodejs об ошибке
5. INFO:Мутация в БД
   1. {Endpoint}:{Параметры, переданные на endpoint}. Подразумеваем, что при создании этого лога результат мутации информации успешен
6. ERROR:Ошибка мутации в БД
   1. {Endpoint}:{Параметры, переданные на endpoint}:{Ошибка}, где Ошибка – стандартный вывод Nodejs об ошибке
7. INFO:Выполнение задачи из журнала задач(schedule)
   1. {Задача}:{Информация по задаче} Подразумеваем, что при создании этого лога результат выполнения задачи успешен. Информация по задаче определяется отдельно для каждой задачи
8. ERROR:Ошибка выполнения задачи из журнала задач
   1. {Задача}:{Информация по задаче}:{Ошибка}, где Ошибка – стандартный вывод Nodejs об ошибке
9. INFO: Отправка письма по электронной почте
   1. {Email куда отправляем}:{С помощью какого сервиса отправляем(основного/резервного)}:{Текст сообщения}
   2. Если отправка сообщений по электронной почте запрещена, то строка не выводится
10. ERROR: Ошибка отправки письма по электронной почте
    1. {Email куда отправляем}:{С помощью какого сервиса отправляем(основного/резервного)}:{Ошибка}

Путь для хранения текстовых файлов начинается с Начальной папки по имени SystemLogs

Должно быть 2 файла, в которые будут писаться соответствующие типы логов

Запись блокируемая, то есть пока программа не закрыта – доступ к текстовым файлам возможен только на чтение с помощью команд, аналогичным tail –f в Unix системах, либо копированием файлов с последующим открытием отдельного инстанса