

# Техническое задание «Архивариус»

## Тезаурус:

МО - медицинская организация

РЭМД - реестр электронных медицинских документов

РИЭМК - региональная интеграционная электронная медицинская карта

SOAP - простой протокол доступа к объектам (условно что-то вроде альтернативы REST). При необходимости изучить технологию самостоятельно.

Необходимо написать бэкэнд приложения для отправки документов в РЭМД с привязкой документов к пациентам.

## Макет фронта:

Веб-портал «Архивариус»

Выйти

Медицинская организация

СНИЛС

Искать в РИЭМК

Фамилия

Имя

Отчество

Дата рождения

Полис ОМС

Номер карты

Выбрать файл...

Выбран файл: Scan001

Описание файла

Подписать и отправить в РЭМД

Статус отправки в РЭМД: Отправлен успешно

## Примерный сценарий использования

1. Сотрудник МО открывает страницу архивариуса и авторизуется

2. Выбирает свою МО из выпадающего списка
3. Вводит СНИЛС и фамилию пациента, по которому хочет отправить документ
4. Нажимает кнопку «Искать в РИЭМК». (Фронт отправляет в бэк запрос /iemk, бэк отправляет точно такой же запрос по адресу сервиса iemk и возвращает ответ от сервиса iemk фронту.)
  1. Если единственный пациент с данными СНИЛС и фамилией найдены в РИЭМК, то данные по ним подставляются в форму
  2. Если найдено несколько пациентов по введенным данным, то выпадает список, из которого сотрудник МО выбирает нужного
  3. Если не найдено ни одного пациента по введенным данным, то выводится соответствующее информационное сообщение и сотрудник МО вводит данные самостоятельно
5. Нажимает кнопку «Выбрать файл» и прикрепляет файл, который хочет отправить
6. Заполняет поле «Описание файла»
7. Нажимает кнопку «Подписать и отправить в РЭМД». Файл отправляется фронтом в РЭМД и статус отправки вместе с данными из формы отправляются в метод /data, откуда сохраняются в БД.

Пользователями будут сотрудники МО.

## **БД**

Сервис будет работать из под пользователя archi  
flyway скрипты будут выполняться из под пользователя postgres  
Скрипты на создание и заполнение имеющихся таблиц в прикреплении

## **REST методы для реализации**

На данный момент необходимо реализовать основные методы бэкэнда без механизма авторизации.

По описанию api можно сгенерировать базу для будущих классов REST контроллеров <https://editor.swagger.io> (практически готовый проект) и классы dto запросов/ответов.

Примечание: описание api найдете в прикреплении в корневом каталоге

### **/user**

Метод, возвращающий данные о пользователе по его снлсу (во фронте пока не будет задействован, но необходим к реализации)

Метод является проксирующим и интегрирует бэкэнд с SOAP сервисом пользователей (RegUserService)

Для интеграции необходимо будет по xsd и wsdl сгенерировать объекты запросов/ответов для клиентского класса сервиса. Сгенерировать можно

с использованием maven плагина `org.jvnet.jaxb2.maven2.maven-jaxb2-plugin` (или аналогичным)

SOAP клиент можно реализовать с использованием класса `org.springframework.ws.client.core.support.WebServiceGatewaySupport` (spring-ws)

Можно генерировать классы для взаимодействия и использовать SOAP клиент свои по желанию. Один из примеров использования вышеуказанных инструментов: <https://howtodoinjava.com/spring-boot/spring-soap-client-webservicetemplate/>

Вызывать необходимо метод `getUser` с переданным СНИЛСом пользователя и возвращать в результате работы объект с указанными в описании api полями. Адрес тестового сервиса: <https://cas-test.hostco.ru/RegUserService/services/RegUserService?wsdl> но можно и по wsdl + xsd поднять свой мок с помощью SoapUI. с помощью этого же SoapUI можно отправлять запросы к SOAP сервисам (в т.ч. к RegUserService).

Примечание: Примеры запроса и ответа, wsdl и xsd к RegUserService в прикреплении в каталоге RegUserService. Поле `userId` в ответе заполнять СНИЛСом пользователя из ответа (поле `//ns2:getUserResponse[1]/ns1:SNILS[1]/text()[1]` в ответе `getUser`). `token` для доступа к сервису указан в примере запроса.

### **/mo**

Метод, возвращающий список МО для выпадающего списка выбора МО

Метод должен возвращать описанные в api сущности по записям из таблицы

только записи с максимальным значением поля `version` и значением поля `is_shown=true`

Скрипт для создания и заполнения таблицы в прикреплении. Данные скрипты нужно добавить в flyway и скрипты на создание всех таблиц так же желательно положить в flyway. Создавать новые объекты в БД в рамках написания метода не требуется.

### **/data**

метод, сохраняющий переданные данные в БД. Должен принимать все данные из запроса и сохранять их в БД в новую таблицу. Создать таблицу, продумать её структуру необходимо самостоятельно. Все поля, кроме поля, в которое сохраняется `userId` должны быть необязательными (при этом поле, в которое сохраняется значение из `userId` не является первичным ключом). В создаваемой таблице по возможности добавить два поля `date_insert` и `date_update`. Заполнять текущими датой и временем при вставке/обновлении записи.

## **/iemk**

метод, осуществляющий поиск пациента по СНИЛС и/или фамилии в ИЕМК. Метод является проксирующим и вызывает точно такой же метод с точно таким же описанием, как в описании api сервиса. Метод сервиса ИЕМК возвращает точно такие же сущности, как описано в api.

### **Технологический стек:**

- java 8 или 11
- maven
- spring-boot 2
- hibernate
- flyway
- postgresql
- публикация кода в github

### **Обращать внимание будем на следующие моменты:**

- Количество реализованных методов и их соответствие ТЗ
- Чистота кода
- Применение ООП и разбиение функционала на классы/методы
- Комментирование кода
- Отсутствие багов и ошибок

### **Инструкция по поднятию и использованию фронта при необходимости:**

1. Установить 14.5 версию node.js ссылка: <https://nodejs.org/en/>
2. Вместе с node.js установится пакетный менеджер npm. npm -v в консоли для проверки версии.
3. Установить ангулар: `npm install -g @angular/cli`
4. В папке с проектом нужно прописать `npm install`
5. Для запуска фронта: `npm start`
6. После того, как в консоли появится «Compiled successfully» можно будет открыть фронт по адресу <http://localhost:4200/> (адрес будет выведен в консоли)

Примечание: Для того чтобы изменить путь до бэка нужно зайти в папку `src/environments`, в файле `environment.ts` изменить `url` на свой. Репозиторий с фронтом в прикреплении.

### **Контакты для связи:**

- Рыбаков Виктор (Разработчик Java) - [https://t.me/siberian\\_love](https://t.me/siberian_love) / <https://vk.com/siberian.love> // Можно писать с любыми вопросами по заданию
- Дарья Хоружевская (HR менеджер компании ХОСТ) - [https://t.me/d\\_khoruzhevskaya](https://t.me/d_khoruzhevskaya)

Сервис с мастер-класса июня 2020 года: <https://github.com/SenotakaiOtoko/gitlab-telegram-integration-1>

Выполненное задание присылать по адресу [d.khoruzhevskaya@hostco.ru](mailto:d.khoruzhevskaya@hostco.ru) с темой «Готовое задание». В задании должна быть ссылка на github репозиторий с исходниками готового задания. В репозитории помимо исходников должна быть инструкция по запуску приложения.