

Постановка на разработку

Документ описывает требования к сохранению и удалению паспортных данных Пользователя.

Автор:	Карпец Александр Игоревич
Количество затраченных часов:	16

- 1. Цели и задачи
- 2. Термины и определения
- 3. Общее описание
 - 3.1 User-story
 - 3.2 Требования
 - 3.2.1 Функциональные требования
 - 3.2.2 Предположения и ограничения
 - 3.2.3 Варианты использования
 - 3.2.3.1 USE CASE для сохранения Документов пользователем в системе
 - 3.2.3.2 USE CASE для удаление Документов Сотрудником
- 4. Права и роли
- 5. Информационная архитектура
 - 5.1 Доменная модель
 - 5.1.1 Агрегат «Документы»
 - 5.1.1.1 Жизненный цикл
 - 5.1.1.2 Атрибутивный состав
 - 5.2 Диаграммы состояний и переходов
 - 5.2.1 Диаграмма последовательности для сохранения Документов
 - 5.2.2 Диаграмма последовательности для удаления Документов
- 6. Пользовательские интерфейсы
 - 6.1 Интерфейсы мобильного приложения (APP)
 - 6.1.1 Интерфейс «Настройки»
 - 6.1.2 Интерфейс «Внесение паспортных данных»
 - 6.1.3 Интерфейс «Данные загружены»
 - 6.1.4 Интерфейс «Неверно указанные данные»
 - 6.1.5 Интерфейс «Ошибка сервиса»
 - 6.2 Интерфейсы бэк-офис (web front)
 - 6.2.1 Интерфейс «Главный экран»
 - 6.2.2 Интерфейс «Неактуальные документы»
 - 6.2.3 Интерфейс «Неактуальные данные отсутствуют»
 - 6.2.4 Интерфейс «Ошибка сервиса»
- 7. Хэндлеры
 - 7.1 Метод получение ИНН POST v1/api/ion/v1/inn
 - 7.1.1 Описание
 - 7.1.2 Входные данные
 - 7.1.3 Выходные данные
 - 7.1.4 Логика работы
 - 7.2 Метод сохранения Документов POST v1/documents/
 - 7.2.1 Описание
 - 7.2.2 Входные данные
 - 7.2.3 Выходные данные
 - 7.2.4 Логика работы
 - 7.3 Метод получения списка неактуальных документов GET v1/baddocuments
 - 7.3.1 Описание
 - 7.3.2 Выходные данные

- 7.3.3 Логика работы
- 7.4 Метод удаление неактуальных Документов DELETE v1/baddocuments
 - 7.4.1 Описание
 - 7.4.2 Выходные данные
 - 7.4.3 Логика работы
- 8. Открытые вопросы
- 9. Дополнительные материалы
 - 9.1 Получение ИНН по паспортным данным
- 10. Идеи на будущее

1. Цели и задачи

1. В мобильное приложение требуется добавить возможность, чтобы каждый пользователь мог указать свои паспортные данные.
2. При указании паспортных данных, они должны проверяться на сервисе .
3. Если данные корректны, и ИНН из сервиса получен, требуется сохранить введенные паспортные данные и ИНН в привязке к пользователю.
4. В один момент времени у пользователя может быть не более одного паспорта.
5. Если возникает необходимость замены паспорта, текущая версия паспорта должна удаляться сотрудником службы поддержки через систему бэк-офиса, давая возможность пользователю мобильного приложения завести новый паспорт.
6. Неудачные попытки ввода паспортных данных не должны сохраняться в системе. Система хранит лишь проверенные/корректные паспорта.
7. После ввода паспортных данных пользователем, если они были приняты системой, по соображениям безопасности у пользователя больше не должно быть возможности просмотреть ранее введенные данные.

2. Термины и определения

№	Термин	
1	Пользователь	Пользователь мобильным приложением (APP)
2	Сотрудник	Сотрудник службы поддержки, работающий в системе бэк-офис.
3	Документы	Паспортные данные и ИНН Пользователя
4	APP	Мобильное приложение в котором работает основная часть пользователей
5	Внешний Сервис	https://service.nalog.ru/inn.do
6	Система	Это система объединяющая в себе бэк-офис, Backend и APP
7	Бэк-офис	Web front, в котором работают сотрудники бэк-офиса
8	Backend	back-end часть Системы с реляционной базой данных

3. Общее описание

3.1 User-story

Как заказчик, я хочу, чтобы Пользователь имел возможность сохранять паспортные данные в системе, чтобы иметь доступ Документам Пользователя.

Как сотрудник службы поддержки, я хочу, иметь возможность удалять устаревшие паспортные данные Пользователя через систему бэк-офиса, чтобы Пользователь имел возможность завести в Систему новые паспортные данные.

3.2 Требования

3.2.1 Функциональные требования

1. Необходимо разработать агрегат “Документы”, предназначенный для хранения Документов Пользователя. Описание агрегата: [Постановка на разработку | 5.1.1 Агрегат “Документы”](#)
2. Необходимо разработать метод POST v1/api/ion/vq/inn для получения ИНН Пользователя со стороны Внешнего Сервиса. Описание метода: [Постановка на разработку | 7.1 Метод получение ИНН POST v1/api/ion/v1/inn](#)
3. Необходимо разработать метод POST v1/documents для создание новой записи в агрегате “Документы”. Описание метода: [Постановка на разработку | 7.2 Метод сохранения Документов POST v1/documents/](#)
4. В APP необходимо разработать и добавить в интерфейс “Настройки” кнопку “Внести паспортные данные”. Описание интерфейса: [Постановка на разработку | 6.1.1 Интерфейс “Настройки”](#)
5. В APP необходимо разработать интерфейс «Внесение паспортных данных». Описание интерфейса: [Постановка на разработку | 6.1.2 Интерфейс «Внесение паспортных данных»](#)
6. В APP необходимо разработать интерфейс «Ваши данные успешно загружены». Описание интерфейса: [Постановка на разработку | 6.1.3 Интерфейс «Данные загружены»](#)
7. В APP необходимо разработать интерфейс «Неверно указанные данные». Описание интерфейса: [Постановка на разработку | 6.1.4 Интерфейс «Неверно указанные данные»](#)
8. В APP необходимо разработать интерфейс «Ошибка сервиса». Описание интерфейса: [Постановка на разработку | 6.1.5 Интерфейс «Ошибка сервиса»](#)
9. Необходимо разработать метод GET v1/baddocuments/ для получение списка ссылок на Пользователей, у которых меньше чем через месяц истекает срок действия паспортных данных. Описание метода: [Постановка на разработку | 7.3 Метод получения списка неактуальных документов GET v1/baddocuments](#)
10. Необходимо разработать метод DELETE v1/baddocuments/ для удаление Документов в которых срок действия паспортных данных истекает меньше чем через месяц. Описание метода: [Постановка на разработку | 7.4 Метод удаление неактуальных Документов DELETE v1/baddocuments](#)
11. В бэк-офисе необходимо разработать и добавить кнопку в интерфейс “Главный экран”. Описание интерфейса: [Постановка на разработку | 6.2.1 Интерфейс «Главный экран»](#)
12. В бэк-офисе необходимо разработать интерфейс «Неактуальные данные». Описание метода: [Постановка на разработку | 6.2.2 Интерфейс «Неактуальные документы»](#)
13. В бэк-офисе необходимо разработать интерфейс «Неактуальные данные отсутствуют». Описание метода: [Постановка на разработку | 6.2.3 Интерфейс «Неактуальные данные отсутствуют»](#)
14. В бэк-офисе необходимо разработать интерфейс «Ошибка сервиса». Описание метода: [Постановка на разработку | 6.2.4 Интерфейс «Ошибка сервиса»](#)

3.2.2 Предположения и ограничения

Идентификатор	Ограничение
CONST-01	На проекте работает 2 отдельные команды разработки: отдельная команда front и отдельная back. (источник: автор задания)
CONST-02	Вопрос авторизации не рассматривается в рамках задания. Например, при вызове GET v1/currentuser полагаем, что пользователя мы можем однозначно определить по сессии. (источник: автор задания)

CONST-03	В рамках задачи UI проектировать нет необходимости. Допустимо общее словесное описание, - что и где Вы бы расположили. (источник: автор задания)
CONST-04	Front и Back взаимодействует по RESTful API. Система находится в промышленной эксплуатации, около 10 000 активных пользователей APP. (источник: автор задания)
CONST-05	На данный момент разработан хэндлер GET v1/currentuser, который позволяет получить данные (никнейм, дату регистрации, номер телефона и т.д.) по текущему пользователю. (источник: автор задания)
CONST-06	Пользователь при регистрации в Системе дал согласие с политикой обработки персональных данных, в которой предусматривалась обработка и хранение Документов Пользователя. (источник: допущение со стороны аналитика)
CONST-07	В Системе паспорт перестаёт считаться действительным за месяц до истечения его срока годности. (источник: допущение со стороны аналитика)
CONST-08	Интеграция с Внешним Сервисом уже настроена и протестирована. Сервис выдал бессрочный access-token. (источник: допущение со стороны аналитика)
CONST-09	В APP есть интерфейс "Настройки". (источник: допущение со стороны аналитика)
CONST-10	В бэк-офисе есть интерфейс "Главный экран". (источник: допущение со стороны аналитика)

3.2.3 Варианты использования

3.2.3.1 USE CASE для сохранения Документов пользователем в системе

Краткое описание: Пользователь свои Документы в Систему (через APP)

Действующие лица: Пользователь и Система.

Предварительное состояние: Пользователь находится в разделе "Настройки"

Запускающие событие: -

Цель: Занесение паспортных данных в Систему.

Результирующее состояние: Система выводит сообщение Пользователю "Ваши данные успешно загружены"

№ шага	Действующее лицо	Шаг/действие	Альтернативный сценарий
1	Пользователь	Нажать на кнопку "Внести паспортные данные"	
2	Система	Отобразить интерфейс «Внесение паспортных данных»	
3	Пользователь	Ввести паспортные данные и нажать кнопку «Сохранить»	В случае, если пользователь передумал вводить данные: 3.1 Пользователь нажимает кнопку "Назад" 3.2 Система отображает интерфейс "Настройки"

4	Система	Проверить , введенные данные.	Если введенные данные неверные: 4.1 Система отобразить интерфейс "Неверно указанные данные" 4.2 Пользователь нажимает кнопку "Спасибо" 4.3 Система отображает интерфейс "Настройки"
5	Система	Отобразить интерфейс "Данные загружены"	Если ошибка сервиса: 5.1 Система отобразить интерфейс "Ошибка сервиса" 5.2 Пользователь нажимает кнопку "Спасибо" 5.3 Система отображает интерфейс "Настройки"
6	Пользователь	Нажать кнопку "Спасибо"	
7	Система	Отобразить интерфейс "Настройки"	

3.2.3.2 USE CASE для удаление Документов Сотрудником

Краткое описание: Сотрудник находясь в Системе (бэк-офис) запускает проверку на актуальность паспортных данных Пользователей в Системе. После чего все неактуальные Документы удаляет.

Действующие лица: Сотрудник и Система.

Предварительное состояние: Сотрудник находится в Системе

Запускающие событие: -

Цель: Удаление не актуальных Документов

Результирующее состояние: Система отображает интерфейс "Неактуальные данные отсутствуют"

№ шага	Действующее лицо	Шаг/действие	Альтернативный сценарий
1	Сотрудник	Нажать на кнопку "Проверка документов пользователей" в интерфейсе "Главный экран"	
2	Система	Выполнить запрос GET v1/documents/	В случае, если запрос пустой: 2.1 Система отобразить интерфейс "Неактуальные данные отсутствуют" 2.2 Сотрудник нажимает кнопку "Спасибо" 2.3 Система отображает интерфейс "Главный экран"
3	Система	Отобразить интерфейс "Неактуальные данные"	В случае, если возникла ошибка:

			<p>3.1 Система отобразить интерфейс “Ошибка сервиса”</p> <p>3.2 Сотрудник нажимает кнопку “Спасибо”</p> <p>3.3 Система отображает интерфейс “Главный экран”</p>
4	Сотрудник	Нажать на кнопку “Удалить документы”	
5	Система	Выполнить запрос DELETE v1/documents/	
6	Система	Отобразить интерфейс “Неактуальные данные отсутствуют”	<p>В случае, если возникла ошибка:</p> <p>6.1 Система отобразить интерфейс “Ошибка сервиса”</p> <p>6.2 Сотрудник нажимает кнопку “Спасибо”</p> <p>6.3 Система отображает интерфейс “Главный экран”</p>
7	Пользователь	Нажать кнопку “Спасибо”	
8	Система	Отобразить интерфейс “Главный экран”	

4. Права и роли

Действие	Роль	Комментарий
Добавление Документов	Пользователь	Сохранение в БД паспортных данных и ИНН через APP.
Удаление Документов	Сотрудник	Удаление из БД паспортных данных и ИНН Пользователей через бэк-офис.

5. Информационная архитектура

5.1 Доменная модель

Примечание: Основное описание доменной модели находится в “Тестовом задании для соискателя”. Там же описание агрегатов Автомат и Пользователь.

5.1.1 Агрегат “Документы”

5.1.1.1 Жизненный цикл

Жизненный цикл и атрибутивный состав наследуется от агрегата Автомат.

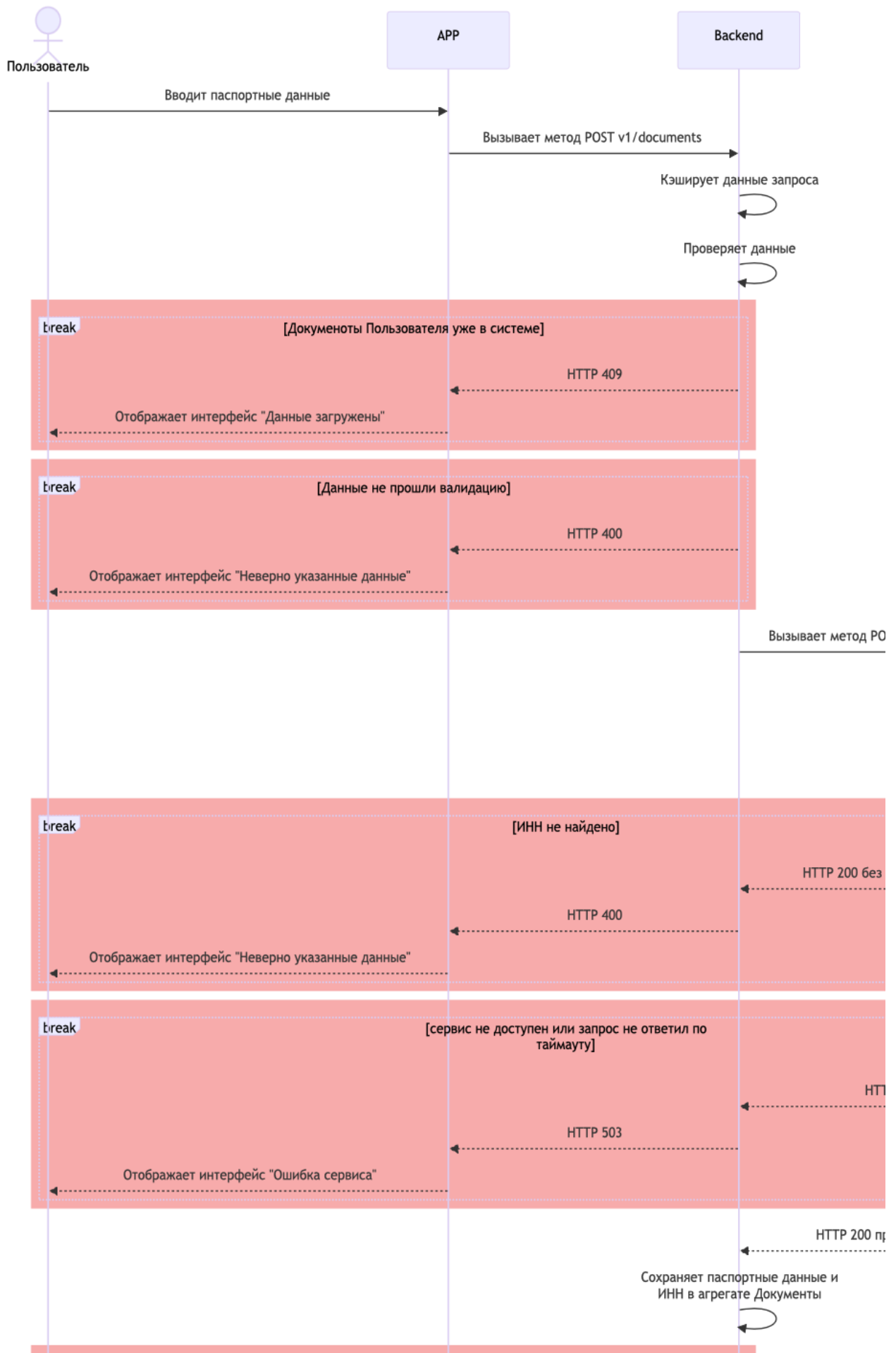
5.1.1.2 Атрибутивный состав

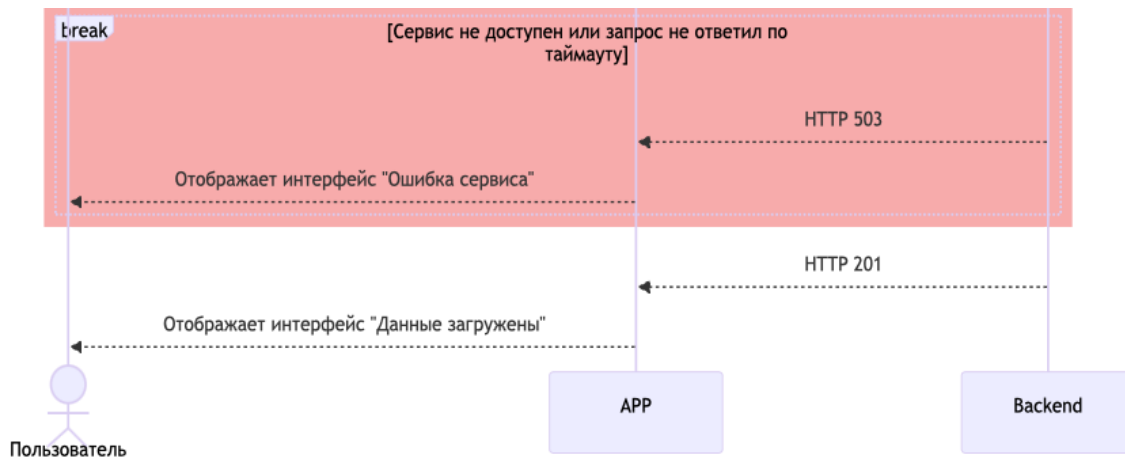
№	Атрибут	Обязательность	Валидатор	Комментарий
---	---------	----------------	-----------	-------------

1	date_of_creation	Да	Timestamp	Дата создания
2	date_of_status_change	Да	Timestamp	Дата изменения статуса
3	status	Да	Enum	Активен, удалён
4	message_of_status	Нет	Text	Сообщение статуса
5	author	Да	Text	Ссылка на экземпляр в агрегате "Пользователь"
6	last_name	Да	Char(50)	Фамилия
7	first_name	Да	Char(50)	Имя
8	middle_name	Нет	Char(50)	Отчество
9	birthday	Да	Date	Дата рождения
10	passport_numbers	Да	Text	Серия и номер паспорта
11	date_of_issue	Да	Date	Дата выдачи
12	inn	Да	Text	ИНН

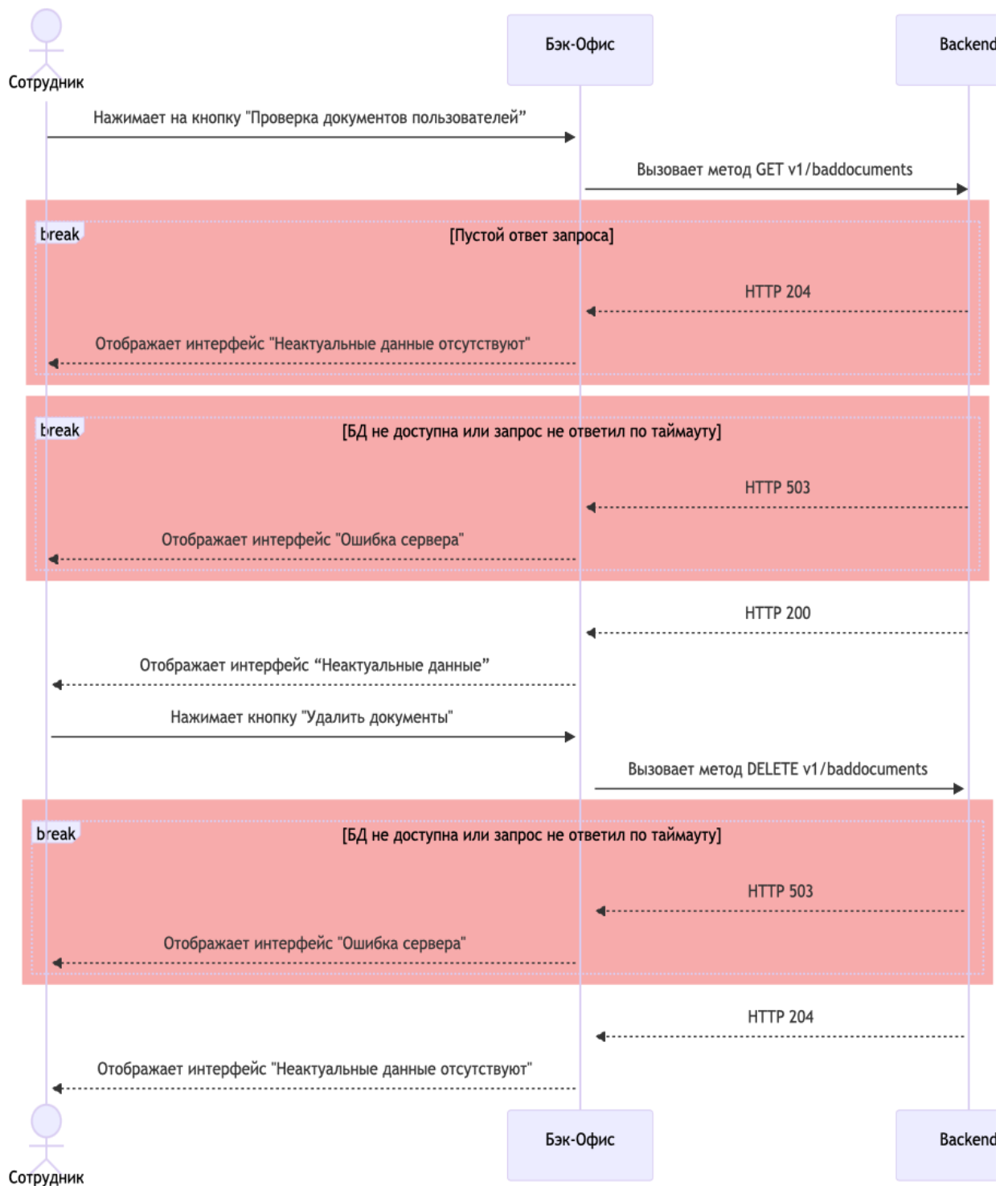
5.2 Диаграммы состояний и переходов [↗](#)

5.2.1 Диаграмма последовательности для сохранения Документов [↗](#)





5.2.2 Диаграмма последовательности для удаления Документов [↗](#)



6. Пользовательские интерфейсы [↗](#)


6.1 Интерфейсы мобильного приложения (APP) [↗](#)

6.1.1 Интерфейс "Настройки" [↗](#)

Кнопка "Внести паспортные данные" расположена в нижней части экрана.

6.1.2 Интерфейс «Внесение паспортных данных»

Интерфейс состоит из трёх частей:

1. Сверху: Заголовок: «Внесение паспортных данных»
2. В середине: Форма для ввода данных состоящая из 6 полей: Имя, Фамилия, Отчество, Дата рождения, дата выдачи паспорта, серия и номер паспорта. Пример образца:  [Тинькофф — финансовые услуги для физических и юридических лиц](#)
3. С низу экрана: Две кнопки: кнопка “Сохранить” и кнопка “Назад”

6.1.3 Интерфейс «Данные загружены»

Интерфейс состоит из двух частей:

1. Сверху: Заголовок «Данные загружены»
2. Под заголовком кнопка “Спасибо”

6.1.4 Интерфейс «Неверно указанные данные»

Интерфейс состоит из двух частей:

1. Сверху: Заголовок «Неверно указанные данные, проверьте и попробуйте ввести снова»
2. Под заголовком кнопка “Спасибо”

6.1.5 Интерфейс «Ошибка сервиса»

Интерфейс состоит из двух частей:

1. Сверху: Заголовок «Ошибка сервиса. Попробуйте ввести данный позже»
2. Под заголовком кнопка “Спасибо”

6.2 Интерфейсы бэк-офис (web front)

6.2.1 Интерфейс «Главный экран»

Кнопка «Проверка документов пользователей» расположена слева в нижней части экрана.

6.2.2 Интерфейс «Неактуальные документы»

Интерфейс состоит из двух частей:

1. Сверху: Заголовок «Список пользователей паспортные данные которых истекают меньше чем через месяц»
2. Под заголовком список который отображает пользователей по запросу GET
3. Под списком кнопка “Удалить документы”

6.2.3 Интерфейс «Неактуальные данные отсутствуют»

Интерфейс состоит из двух частей:

1. Сверху: Заголовок «Неактуальных документы в системе отсутствуют»
2. Под заголовком кнопка “Спасибо”

6.2.4 Интерфейс «Ошибка сервиса»

Интерфейс состоит из двух частей:

1. Сверху: Заголовок «Ошибка сервиса. Попробуйте позже»
2. Под заголовком кнопка “Спасибо”

7. Хэндлеры

7.1 Метод получение ИНН POST v1/api/ion/v1/inn

7.1.1 Описание

Метод вызывается для получения информации об ИНН Пользователя по паспортным данным из Внешнего Сервиса. В качестве формата обмена данными используется JSON. Рекомендуемая частота вызова процедуры: 1 раз в 5 сек и реже. Метод можно вызвать только после получения паспортных данных от пользователя.

Запрос направляется во Внешнюю Систему с использованием технологии синхронного взаимодействия. Схематическое представление взаимодействия при запросе представлено в "Диаграмме последовательности для сохранения Документов".

Участники информационного обмена используют POST метод с URL [/ion/v1/inn](#). В заголовке REST запроса указывается токен доступа: accessToken. Важно: accessToken должен быть закодирован в base64.

7.1.2 Входные данные

Содержание	Тип	Обязательность	Описание
data	Struct	Да	Список информации о Пользователе
· id	String(50)	да	Уникальный идентификатор физического лица Важно! В качестве id должен использоваться GUID. Например, fe6ed552-a902-44e2-8170-991ce43e5bb0
· lastName	String(60)	Да	Фамилия Пользователя
· firstName	String(60)	Да	Имя Пользователя
· secondName	String(60)	Нет	Отчество Пользователя
· passportSeries	String(30)	Да	Формат данных: "XX XX", где X - цифра от 0 до 9. Между парами цифр необходим пробел
· passportNumber	String(30)	Да	Формат данных: 'XXXXXX', где X - цифра от 0 до 9. Допустимо указание 7 цифр
· birthday	String(10)	Да	Дата рождения. Формат уууу-mm-dd
· documentCode	String(5)	Да	Должен стоять код: '21'

7.1.3 Выходные данные

Содержание	Тип	Обязательность	Описание
requestId	String(50)	Да	Идентификатор сообщения. Совпадает с X-Request_Id назначенный системой автоматически, при создании запроса.
requestType	String(6)	Да	Тип ответа: SINGLE
responseDocumentItems	Struct		Сведения об ИНН
· id	String(50)	Да	Уникальный идентификатор физического лица в рамках одного запроса для определенного участника ИО с уникальным идентификатором запроса. В формате GUID.
· inn	String(12)	Да	ИНН Пользователя
· businessError	Struct	Да	Содержит информацию о возникших ошибках при обработке запроса. Если ошибок нет, то передается значение "businessError": null
o code	String(255)	Нет	Код ошибки
o message	String(255)	Нет	Описание ошибки

7.1.4 Логика работы

После получение запроса во Внешнем Сервисе, запрос проходит следующие проверки:

1. Если запрос не проходит Форматно-Логический контроль, то фиксируется ошибка и направляется сообщениес HTTP кодом запроса 200:

```
1 {
2   "requestId": "d8b5b32a-9fba-4537-a2f4-e7642605341a",
3   "requestType": "SINGLE",
4   "responseDocumentItems": [
5     {
6       "id": "",
7       "inn": null,
8       "businessError": {
9         "code": "invalid.data",
```

```

10     "message": "Данные запроса не прошли ФЛК"
11   }
12 }
13 ]
14 }

```

2. Если проверка пройдена, тогда происходит поиск ИНН по указанным в запросе данными о Пользователе. Если ИНН по запрашиваемому Пользователя не найден, тогда приходит ответ с сообщением с HTTP кодом запроса 200:

```

1 {
2   "requestId": "4ad76228-0000-0002-aaaa-eef36d3de35a",
3   "requestType": "SINGLE",
4   "responseDocumentItems": [
5     {
6       "id": "4ad76228-0000-0000-aaaa-eef36d3de35a",
7       "inn": null,
8       "businessError": {
9         "code": "inn.not.found",
10        "message": "Невозможно предоставить ИНН по указанным в запросе сведениям о НП"
11      }
12    }
13  ]
14 }

```

3. При наличии внутренних ошибок Внешний Сервис направляет ответ с HTTP кодом запроса 500. Пример сообщения:

```

1 {
2   "requestId": "4ad13552-7648-4541-a167-0a069f9152d8",
3   "businessError": {
4     "code": "internal.error",
5     "message": "Внутренняя ошибка"
6   }
7 }

```

4. Если данные об ИНН Пользователя найдены, тогда в ответе приходит с сообщением с HTTP кодом запроса 200. Пример сообщения:

```

1 {
2   "requestId": "4ad76228-0000-0002-aaaa-eef36d3de35a",
3   "requestType": "SINGLE",
4   "responseDocumentItems": [
5     {
6       "id": "4ad76228-0000-0000-aaaa-eef36d3de35a",
7       "inn": "123456789012",
8       "businessError": null
9     }
10  ]
11 }

```

7.2 Метод сохранения Документов POST v1/documents/

7.2.1 Описание

Метод вызывается для сохранения Документов пользования в систему. Метод начинается с отправки паспортных данных Пользователем. Схематическое представление взаимодействия при запросе представлено в “Диаграмме последовательности для сохранения Документов”. Успешный вызов метода добавляет экземпляр в агрегат “Документы”.

7.2.2 Входные данные

№	Атрибут	Тип	Обязательность	Комментарий
1	date_of_creation	Timestamp	Да	Дата создания
2	date_of_status_change	Timestamp	Да	Дата изменения статуса
3	status	Enum	Да	Должен стоять статус "Активен"
4	last_name	Char(50)	Да	Фамилия
5	first_name	Char(50)	Да	Имя
6	middle_name	Char(50)	Нет	Отчество
7	birthday	Date	Да	Дата рождения
8	passport_numbers	Text	Да	Серия и номер паспорта
9	date_of_issue	Date	Да	Дата выдачи

7.2.3 Выходные данные

№	Название	Тип	Обязательность	Комментарий
1	Meta	Struct	Да	Результат вызова метода
1.1	status	string	Да	Код результата
1.2	description	string	Да	Пользовательское сообщение

7.2.4 Логика работы

1. После получения запроса от пользователя, Система кэширует запрос и ссылку на экземпляр агрегата "Пользователь", где хранятся данные Пользователя.
2. Если документы данного Пользователя уже сохранены в агрегате "Документы", то завершать вызов метода ошибкой с кодом 409, meta.status=="CONFLICT", meta.description=="Документы уже в базе";
3. Если проверка на валидность атрибутов в запросе не прошла, то завершать вызов метода ошибкой с кодом 400, meta.status=="ERROR", meta.description=="Некорректные данные";
4. Если проверки валидности названия команды успешно пройдены, необходимо выполнить метод POST v1/api/ion/v1/inn, чтобы получить данные ИНН Пользователя;
5. Если Внешний Сервис не доступен или запрос не ответил по таймауту, то завершать вызов метода ошибкой с кодом 503, meta.status=="ERROR", meta.description=="Ошибка сервиса";
6. Если Внешний Сервис прислал ошибку с кодом 500 или другой неизвестной ошибкой, то завершать вызов метода ошибкой с кодом 503, meta.status=="ERROR", meta.description=="Ошибка сервиса";
7. Если Внешний Сервис прислал ошибку с кодом 200 в котором отсутствует ИНН ("inn": null), то завершать вызов метода ошибкой с кодом 400 meta.status=="ERROR", meta.description=="Некорректные данные";
8. Если запрос во Внешний Сервис прошёл успешно, то необходимо сохранить паспортные данные, ссылку на экземпляр в агрегате "Пользователь и ИНН в качестве экземпляра в агрегате "Документы";

9. Если сервис не доступен или запрос не ответил по таймауту, то завершать вызов метода ошибкой с кодом 503
meta.status=='ERROR', meta.description=='Ошибка сервиса';
10. Если добавление Документов успешно выполнено, то завершать вызов метода успешным ответом с кодом 201,
meta.status=='CREATED', meta.description=='Данные загружены'.

7.3 Метод получения списка неактуальных документов GET v1/baddocuments [↗](#)

7.3.1 Описание [↗](#)

Метод предназначен для получения списка ссылок на Пользователей у которых срок действия паспортных данных истекает меньше чем через 30 дней. Иными словами всех Пользователей, чей возраст находится между 19 годами 11 месяцами и 20 годами ровно, или между 44 годами 11 месяцами и 45 годами ровно. Метод вызывается Сотрудником из системы бэк-офис. Схематическое представление взаимодействия при запросе представлено в “Диаграмме последовательности для удаления Документов”. Успешный вызов метода выводит список ссылок на Пользователей из агрегата “Документы”. Запрос может быть пустым.

7.3.2 Выходные данные [↗](#)

№	Название	Тип	Обязательность	Комментарий
1	Meta	Struct	Да	Результат вызова метода
1.1	status	string	Да	Код результата
1.2	description	string	Да	Пользовательское сообщение
2	Data	Struct	Нет	
2.1	authors	string	Нет	Ссылки на Пользователей

7.3.3 Логика работы [↗](#)

1. Система выполняет запрос список ссылок на Пользователей из агрегата “Документы”, у которых возраст находится между 19 годами 11 месяцами и 20 годами ровно, или между 44 годами 11 месяцами и 45 годами ровно;
2. Если запрос пустой (пустой результат SELECT), то завершить вызов метода ошибкой с кодом 204, meta.status=='NO CONTENT', meta.description=='Неактуальные данные не обнаружены';
3. Если БД не доступна или запрос не ответил по таймауту, то завершать вызов метода ошибкой с кодом 503, meta.status=='ERROR', meta.description=='Ошибка сервиса';
4. Если список успешно получен, то завершать вызов метода успешным ответом с кодом 200, meta.status=='OK', meta.description=='Неактуальные данные', data.authors==result.

7.4 Метод удаление неактуальных Документов DELETE v1/baddocuments [↗](#)

7.4.1 Описание [↗](#)

Метод предназначен для удаление экземпляров из агрегата Документы, полученных в результате применения метода GET v1/baddocuments. Метод вызывается Сотрудником из системы бэк-офис. Схематическое представление взаимодействия при запросе представлено в “Диаграмме последовательности для удаления Документов”. Успешный вызов метода экземпляры у которых срок действия паспортных данных истекает меньше чем через 30 дней из агрегата “Документы”. Иными словами всех Пользователей, чей возраст находится между 19 годами 11 месяцами и 20 годами ровно, или между 44 годами 11 месяцами и 45 годами ровно.

7.4.2 Выходные данные

№	Название	Тип	Обязательность	Комментарий
1	Meta	Struct	Да	Результат вызова метода
1.1	status	string	Да	Код результата
1.2	description	string	Да	Пользовательское сообщение

7.4.3 Логика работы

1. Система выполняет запрос на удаление всех экземпляров из агрегата “Документы”, чей возраст находится между 19 годами 11 месяцами и 20 годами ровно, или между 44 годами 11 месяцами и 45 годами ровно;
2. Если БД не доступна или запрос не ответил по таймауту, то завершать вызов метода ошибкой с кодом 503, `meta.status=='ERROR', meta.description=='Ошибка сервиса'`;
3. Если запрос успешно выполнен, то завершить вызов метода успешным ответом с кодом 204, `meta.status=='NO CONTENT', meta.description=='Неактуальные данные удалены'`.

8. Открытые вопросы

1. Как возникает необходимость замены паспорта? В какой момент? Кто источник? Это должна проверять система или сотрудник бэк-офиса или об этом должен уведомить Пользователь?
2. Как Пользователь узнает, что его паспортные данные удалены?
3. Интеграционное взаимодействие с Внешним Сервисом уже настроено? Или необходимо отправить запрос на получение master-token?
4. Почему паспортные данные должен удалять Сотрудник, а не Система автоматически?
5. Уже разработана Политикой обработки персональных данных, в которой предусматривалась обработка и хранение Документов Пользователя?
6. Какие интерфейсы уже реализованы?

9. Дополнительные материалы

9.1 Получение ИНН по паспортным данным

Не нашёл прямой интеграции для получения ИНН по паспортным данным с service.nalog.ru/inn.do

У Тинькова есть похожий сервис: <https://www.tinkoff.ru/inn/> И есть возможность интеграции: <https://developer.tinkoff.ru/docs/api/get-api-v1-individual-documents-inn>

<https://tinkoff.github.io/tinkoff-id/inn/>

Есть интеграция через Главный научный инновационный внедренческий центр (ГНИВЦ) <https://www.gnivc.ru/software/fnspo/inn/>

В описание хэндлера **POST v1/api/ion/v1/inn** я использовал данные ГНИВЦ.

10. Идеи на будущее

1. Автоматизировать проверку и удаление устаревших Документов. А также уведомление Пользователя о том, что ему стоит добавить новые паспортные данные.