# 1. Описание мастер-системы

## 1.1 Используемые сущности

**1.1.1. Карточка клиента (ClientCard)** – содержит полную информацию о клиенте. Type Object.

Атрибуты:

1. Идентификатор клиента (ClientCardId) – уникальный номер клиента в мастер-системе.
2. Имя (Name). Строка до 250 символов. Обязательный атрибут при создании.
3. Фамилия (Surname). Строка до 250 символов. Обязательный атрибут при создании.
4. Отчество (ParentName). Строка до 250 символов.
5. Телефон (Phone). Строка до 20 символов.
6. Дата рождения (DateBirth). Дата в формате DD.MM.YYYY. Обязательный атрибут при создании.
7. ДУЛ (PersonDocs). Хотя бы 1 обязательный при создании. Отношение 1 ко многим (у одного экземпляра ClientCardId может быть несколько PersonDocs.
8. Адрес (Address). Строка до 10000 символов.
9. Признак согласия на обработку персональных данных (PersFlag). Boolean. Обязательный при создании.
10. Признак согласия на получение рекламы (AdvFlag). Boolean.

**1.1.2. Документ, удостоверяющий личность (ДУЛ) (PersonDoc)**

1. Идентификатор ДУЛ в мастер-системе (PersonDocID).
2. Тип документа (TypeDoc). Ссылка на DocTypes. Обязательный атрибут.
3. Серия документа (SeriaDoc). Строка до 10 символов. Обязательный атрибут.
4. Номер документа (NumDoc). Строка до 10 символов. Обязательный атрибут.
5. Кем выдан (Issuer). Строка до 200 символов. Обязательный атрибут.
6. Дата выдачи документа (IssueDate). Дата в формате DD.MM.YYYY. Обязательный атрибут.
   * 1. **Типы ДУЛ (DocTypes)**
7. Идентификатор типа ДУЛ в мастер-системе (DocTypeID).
8. Код типа документа (DocCode). Обязательный атрибут.
9. Серия документа (SeriaDoc). Строка до 10 символов. Обязательный атрибут.
10. Описание (Descr). Строка до 50 символов.

В рассматриваемом примере используются:

01 – Паспорт

02 – Зарубежный паспорт

03 – Паспорт моряка

**1.1.4. Критерии поиска клиентов (ClientCriteria)** – ограниченный набор атрибутов, используемый для поиска. Type Object.

Атрибуты:

1. Имя (Name).
2. Фамилия (Surname).
3. Отчество (ParentName).
4. Документ, удостоверяющий личность (PersonDoc).

**1.1.5. RespStr – структура ответов при ошибках и успешном выполнении операций**

1. Код ошибки
2. Тип ошибки. Response\_on\_error, Response\_on\_success.
3. Описание ошибки

## 1.2 Методы API

**1.2.1. Общая информация**

Мастер-система предоставляет API для осуществления операций с клиентами и содержит:

1.2.1. внутреннюю логику по проверке полноты, корректности, непротиворечивости данных, в частности:

1.2.1.1. При создании новых ClientCards осуществляется проверка, что такой клиент уже существует в системе. Перечень проверок подразделяется на мягкие и жесткие проверки:

1.2.1.1.1. Жесткая проверка на совпадение реквизитов ДУЛ (возвращается ошибка, клиент не создается),

1.2.1.1.2. Мягкая проверка на полное совпадение ФИО и даты рождения (клиент создается, возвращается сообщение с указанием совпадающих атрибутов),

1.2.1.1.3. Иные проверки, реализованные в мастер-системе.

1.2.2. Внутреннюю логику поиска клиентов по набору критериев (поиск по маске имени, фамилии, отчества, реквизитов ДУЛ, независимость от регистра, возможность использования специальных символов для поиска (“%”, “\*” и т.п.), выдача списка найденных клиентов при одновременном соответствии нескольких атрибутов указанным для поиска маскам и т.п.).

**1.2.2. Перечень и общее описание методов**

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование метода** | **Входящие параметры** | **Возвращаемые значения** | **Общее описание** |
| Post(ClientCard) | ClientCard | * ID созданной ClientCard в мастер-системе (ClientCardId) * либо Null с кодом ошибки * либо ClientCardId с предупредительным сообщением о совпадении атрибутов с атрибутами каких-либо существующих клиентов | Создает новые ClientCard при условии прохождения всех жестких проверок, реализованных в мастер-системе. При создании клиента осуществляются проверки, что такой клиент уже зарегистрирован.  При жесткой проверке проверяется одновременное совпадение следующих полей:   1. Серия 2. Номер 3. Дата 4. Тип дул   В случае непрохождения жесткой проверки возвращается ошибка «Клиент не создан».  При мягкой проверке проверяется полное совпадение ФИО и даты рождения. В случае непрохождения мягкой проверки клиент создается в мастер-системе, но  формируется предупредительное сообщение с указанием всех CleintCard с совпавшими атрибутами. |
| Retrive(ClientCardID) | ClientCardId | * ClientCard с полным набором полей * либо Null с кодом ошибки | Осуществляет поиск ClientCard по ClientCardID |
| Get(ClientCriteria) | ClientCriteria | * Одна либо несколько (множество) ClientCard * либо Null с кодом ошибки | Осуществляет поиск ClientCard по набору критериев (ClientCriteria) по маскам значений атрибутов независимо от регистра. Предусматривает возможность использования специальных символов для поиска (“%”, “\*” и т.п.). Отбирает ClientCards при одновременном соответствии нескольких атрибутов указанным для поиска маскам. |

# 2. Описание микросервиса-адаптера

## 2.1 Используемые сущности

Микросервис-адаптер использует те же сущности, что и мастер-сервис.

## 2.2 Методы API

**1.2.1. Общая информация**

Микросервис-адаптер предоставляет API для осуществления операций с клиентами, осуществляет проверки:

1. входящих параметров на соответствие протоколу,
2. обязательных параметров,
3. http-статусов, полученных от API мастер-системы.

**1.2.2. Перечень и общее описание методов**

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование метода** | **Входящие параметры** | **Возвращаемые значения** | **Общее описание** |
| Post(ClientCard) | ClientCard | * http status 200: успешное выполнение и ID созданной ClientCard в мастер-системе (ClientCardId) * либо Null с кодом ошибки * либо ClientCardId с предупредительным сообщением о совпадении атрибутов с атрибутами каких-либо существующих клиентов | Осуществляет проверки:   * входящих параметров на соответствие протоколу, * обязательных параметров.   При успешном прохождении проверок осуществляет вызов метода Post(ClientCard) мастер-системы (описан в п.1.2.2).  Осуществляет проверки http-статусов, полученных от API мастер-системы.  Возвращает значения, указанные в колонке «Возвращаемое значение». |
| Retrive(ClientCardID) | ClientCardId | * http status 200: успешное выполнение и ClientCard с полным набором полей (информация о найденом клиенте ClientCard) * либо Null с кодом ошибки | Осуществляет проверки:   * входящего параметра на соответствие протоколу, * обязательного наличия параметра ClientCardID.   При успешном прохождении проверок осуществляет вызов метода Retrive(ClientCardID) мастер-системы (описан в п.1.2.2). Возвращает значения, указанные в колонке «Возвращаемое значение». |
| Get(ClientCriteria) | ClientCriteria | * http status 200: успешное выполнение и одна либо несколько (множество) ClientCard (информация о найденных клиентах ClientCard) * либо Null с кодом ошибки | Осуществляет проверки:   * входящих параметров на соответствие протоколу, * обязательных параметров.   При успешном прохождении проверок осуществляет вызов метода Get(ClientCriteria) мастер-системы (описан в п.1.2.2).  Возвращает значения, указанные в колонке «Возвращаемое значение». |

**1.2.3. Описание внутренней логики методов сервиса в виде пошаговых сценариев**

**1.2.3.1. Post(ClientCard)**

**Основной поток:**

1. Проверка наличия всех обязательных параметров (обязательные параметры указаны в п. 1.1.1.).

1.1. Если проверка пройдена – переход к следующему шагу основного потока сценария.

1.2. Если проверка не пройдена – переход к Альтернативному потоку А.

2. Проверки входящих параметров на соответствие протоколу.

2.1. Если проверка пройдена – переход к следующему шагу основного потока сценария.

2.2. Если проверка не пройдена – переход к Альтернативному потоку B.

3. При успешном прохождении всех проверок осуществляется вызов метода Post(ClientCard) мастер-системы (описан в п.1.2.2).

4. Проверка http-статусов, полученных от API мастер-системы.

4.1. Если http status 200: успешное выполнение, то осуществляется переход к следующему шагу основного потока сценария.

4.2. Если http status <> 200 - переход к Альтернативному потоку C.

5. Проверка наличия предупредительного сообщения в структуре ответа мастер-системы.

6. Возвращает http status, полученный от мастер – системы, и ID созданной ClientCard в мастер-системе (ClientCardId). При наличии в структуре ответа мастер-системы предупредительного сообщения возвращает данное предупредительное сообщение.

7. При http status 500: внутренняя ошибка сервиса – переход к альтернативному потоку D.

**Альтернативные потоки:**

***А. Не пройдена проверка наличия всех обязательных параметров.***

1. Возврат ошибки с кодом и описанием ошибки («Не указаны обязательные параметры»).

2. Выход (Exit).

***B. Не пройдена проверка входящих параметров на соответствие протоколу.***

1. Возврат ошибки с кодом и описанием ошибки.

2. Выход (Exit).

***C. http-статус, полученный от API мастер-системы, <> 200.***

***D. http status 500: внутренняя ошибка сервиса:***

1. Возврат ошибки с кодом ошибки и описанием ошибки.

**1.2.3.2. Retrive(ClientCardID)**

1. Проверка наличия обязательного параметра (ClientCardID).

1.1. Если проверка пройдена – переход к следующему шагу основного потока сценария.

1.2. Если проверка не пройдена – переход к Альтернативному потоку А.

2. Проверки входящего параметра на соответствие протоколу.

2.1. Если проверка пройдена – переход к следующему шагу основного потока сценария.

2.2. Если проверка не пройдена – переход к Альтернативному потоку B.

3. При успешном прохождении всех проверок осуществляется вызов метода Retrive(ClientCardID) мастер-системы (описан в п.1.2.2).

4. Проверка http-статусов, полученных от API мастер-системы.

4.1. Если http status 200: успешное выполнение, то осуществляется переход к следующему шагу основного потока сценария.

4.2. Если http status <> 200 - переход к Альтернативному потоку C.

5. Возвращает http status, полученный от мастер – системы, и информацию о найденном клиенте (структуру данных ClientCard, возвращенную мастер-системой).

6. При http status 500: внутренняя ошибка сервиса – переход к альтернативному потоку D.

**Альтернативные потоки:**

***А. Не пройдена проверка наличия обязательного параметра:***

1. Возврат ошибки «Не указаны обязательные параметры»

2. Выход (Exit).

***B. Не пройдена проверка входящего параметра на соответствие протоколу:***

1. Возврат ошибки с кодом ошибки.

2. Выход (Exit).

***C. http-статус, полученный от API мастер-системы, <> 200,***

***D. http status 500: внутренняя ошибка сервиса:***

1. Возврат ошибки с кодом ошибки и описанием ошибки.

**1.2.3.3. Get(ClientCriteria)**

1. Проверка наличия всех обязательных параметров (обязательные параметры указаны в п. 1.1.4.).

1.1. Если проверка пройдена – переход к следующему шагу основного потока сценария.

1.2. Если проверка не пройдена – переход к Альтернативному потоку А.

2. Проверки входящих параметров на соответствие протоколу.

2.1. Если проверка пройдена – переход к следующему шагу основного потока сценария.

2.2. Если проверка не пройдена – переход к Альтернативному потоку B.

3. При успешном прохождении всех проверок осуществляется вызов метода Get(ClientCriteria) мастер-системы (описан в п.1.2.2).

4. Проверка http-статусов, полученных от API мастер-системы.

4.1. Если http status 200: успешное выполнение, то осуществляется переход к следующему шагу основного потока сценария.

4.2. Если http status <> 200 - переход к Альтернативному потоку C.

5. Возвращает http status, полученный от мастер – системы, и информацию об одном или нескольких найденных клиентах (одну либо несколько ClientCard, возвращенных мастер-системой).

6. При http status 500: внутренняя ошибка сервиса – переход к альтернативному потоку D.

**Альтернативные потоки:**

***А. Не пройдена проверка наличия всех обязательных параметров.***

1. Возврат ошибки с кодом и описанием ошибки («Не указаны обязательные параметры»).

2. Выход (Exit).

***B. Не пройдена проверка входящих параметров на соответствие протоколу.***

1. Возврат ошибки с кодом и описанием ошибки.

2. Выход (Exit).

***C. http-статус, полученный от API мастер-системы, <> 200,***

***D. http status 500: внутренняя ошибка сервиса:***

1. Возврат ошибки с кодом ошибки и описанием ошибки.