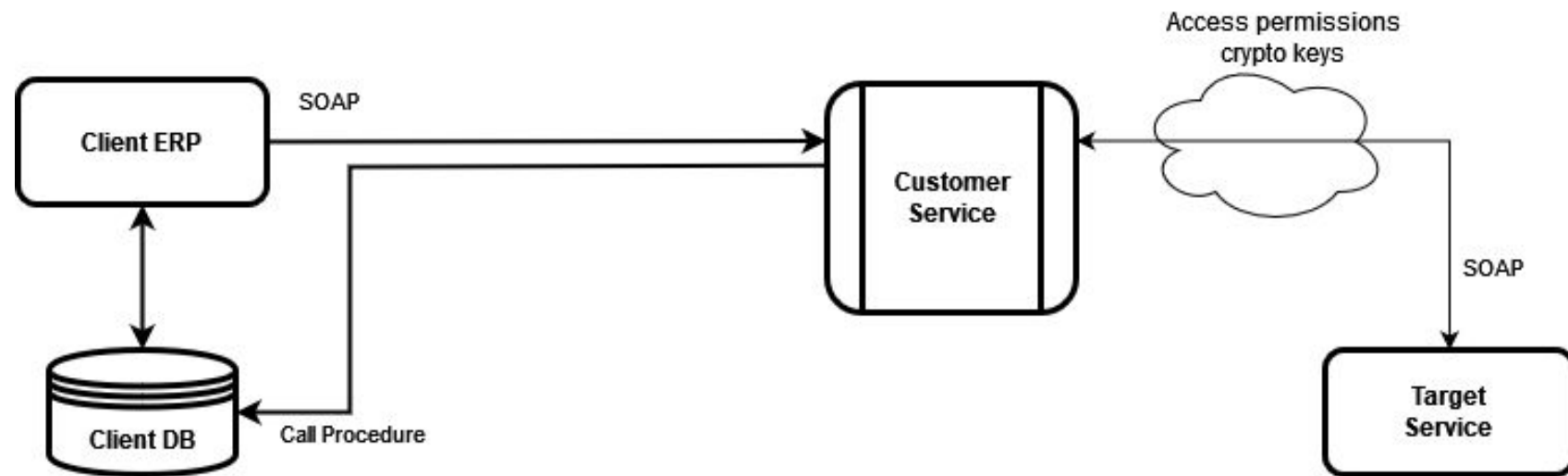


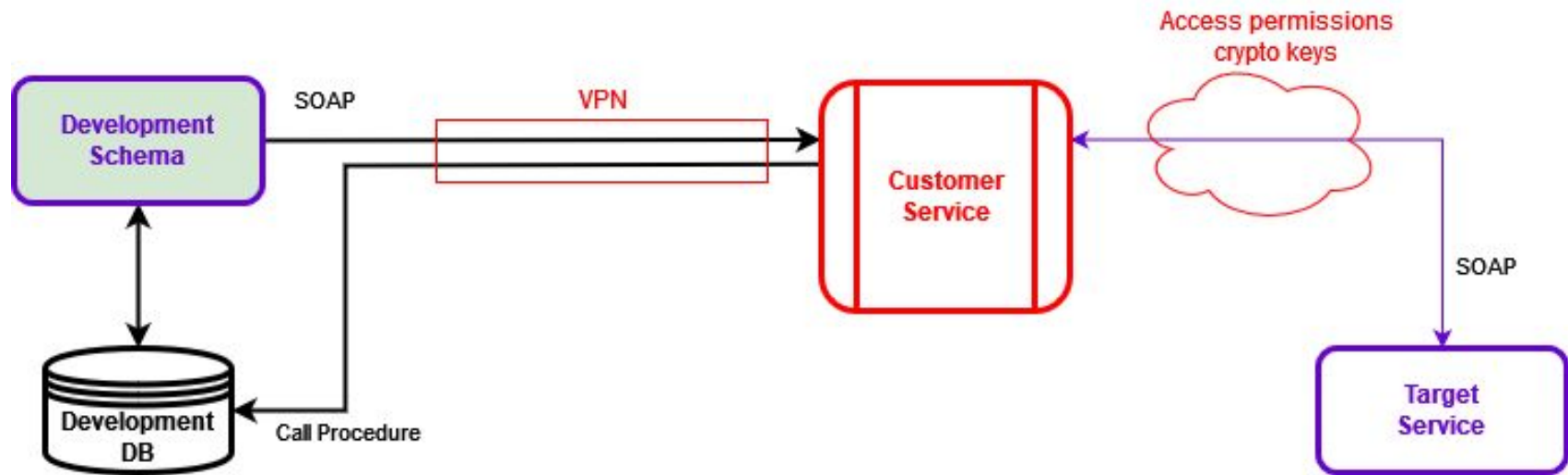
# Эмулятор SOAP сервиса

Spring boot, SOAP, REST, JPA, Ajax

# Схема процесса



# Проблемы №1



# Обычная схема эмуляции

The screenshot displays the SoapUI application interface. On the left, a project tree shows a hierarchy: **projects** > **HelloWorld** > **REST Project 1** > **isna** > **SyncChannelHttpBinding** > **SyncChannelHttpBinding TestSuite** > **SendMessage TestCase** > **Test Steps (1)** > **SendMessage** > **Load Tests (0)** > **Security Tests (0)** > **ISNA\_MockService** > **SendMessage** > **SuccessResponse** > **FailResponse**. The **FailResponse** item is selected.

The main window is divided into two panes. The top pane, titled **ISNA\_MockService**, shows the **Operations** tab with a **SendMessage** operation. The bottom pane, titled **FailResponse**, displays the raw XML of the response:

```
<?xml version='1.0' encoding='UTF-8'>
<soapenv:Envelope xmlns:soapenv='http://schemas.xmlsoap.org/soap/envelope/'>
  <soapenv:Header/>
  <soapenv:Body>
    <typ:SendMessageResponse xmlns:typ='http://schemas.xmlsoap.org/soap/types/'>
      <response>
        <responseInfo>
          <messageId>{messageId}</messageId>
          <responseDate>{responseDate}</responseDate>
          <status>
            <code>26</code>
            <message>Указан некорректный вид операции</message>
          </status>
        </responseInfo>
      </response>
    </typ:SendMessageResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

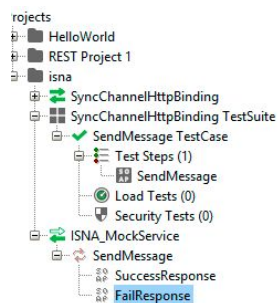
The right pane, titled **SuccessResponse**, displays the raw XML of the response:

```
<?xml version='1.0' encoding='UTF-8'>
<soapenv:Envelope xmlns:soapenv='http://schemas.xmlsoap.org/soap/envelope/'>
  <soapenv:Header/>
  <soapenv:Body>
    <typ:SendMessageResponse xmlns:typ='http://schemas.xmlsoap.org/soap/types/'>
      <response>
        <responseInfo>
          <messageId>{messageId}</messageId>
          <responseDate>{responseDate}</responseDate>
          <status>
            <code>OK</code>
            <message>Message processed successfully</message>
          </status>
        </responseInfo>
      </response>
    </typ:SendMessageResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

At the bottom left, a **Custom Properties** window is open, showing the **MockResponse Properties** table:

Property	Value
Name	FailResponse
Description	
Message Size	656
Encoding	UTF-8

# Проблемы №2



Custom Properties	
MockResponse Properties	
Property	Value
Name	FailResponse
Description	
Message Size	656
Encoding	UTF-8

The screenshot displays the SoapUI interface with two XML response windows. The top window, titled 'SuccessResponse', shows a SOAP message with a status of 'OK' and the text 'Message processed successfully'. The bottom window, titled 'FailResponse', shows a SOAP message with a status of '26' and the text 'Указан некорректный вид операции'. A red rectangle highlights a portion of the 'FailResponse' XML body, specifically the data element containing a timestamp and a long alphanumeric string.

**SuccessResponse XML:**

```
<?xml version='1.0' encoding='UTF-8'?>
<soap:Envelope xmlns:soap='http://schemas.xmlsoap.org/soap/envelope/'>
  <soap:Header/>
  <soap:Body>
    <typ:SendMessageResponse xmlns:typ='http://schemas.xmlsoap.org/soap/envelope/'>
      <response>
        <responseInfo>
          <messageId>{messageId}</messageId>
          <responseDate>{responseDate}</responseDate>
          <status>
            <code>OK</code>
            <message>Message processed successfully</message>
          </status>
        </responseInfo>
        <responseData>
          <data>
            <ICDATA{4:
              :20:0818006500009D24
              :12:400
              :77E:FORMS/A1C/230317/Подтверждение об открытии и закрытии ба
              /ACCOUNT      ,KZKZ      22204398T17486/00/2/220615/881762
            </data>
          </responseData>
        </response>
      </typ:SendMessageResponse>
    </soap:Body>
  </soap:Envelope>
```

**FailResponse XML:**

```
<?xml version='1.0' encoding='UTF-8'?>
<soap:Envelope xmlns:soap='http://schemas.xmlsoap.org/soap/envelope/'>
  <soap:Header/>
  <soap:Body>
    <typ:SendMessageResponse xmlns:typ='http://schemas.xmlsoap.org/soap/envelope/'>
      <response>
        <responseInfo>
          <messageId>{messageId}</messageId>
          <responseDate>{responseDate}</responseDate>
          <status>
            <code>26</code>
            <message>Указан некорректный вид операции</message>
          </status>
        </responseInfo>
        <responseData>
          <data>
            <ICDATA{4:
              :20:0818006500009D24
              :12:400
              :77E:FORMS/A1C/230317/Подтверждение об открытии и закрытии ба
              /ACCOUNT      ,KZKZ      22204398T17486/00/2/220615/881762
            </data>
          </responseData>
        </response>
      </typ:SendMessageResponse>
    </soap:Body>
  </soap:Envelope>
```

# Инструмент, который бы не помешал

1. Возможность анализировать входные текстовые данные и формировать ответ в зависимости от них.
2. Возможность настраивать ответы

# Инструмент, который бы не помешал

1. Возможность анализировать входные текстовые данные и формировать ответ в зависимости от них.
2. Возможность настраивать ответы

## Возможности в среде разработки

1. Запуск приложений на Java 11
2. Возможность слушать необходимый http порт любым приложением

# Инструмент, который бы не помешал

1. Возможность анализировать входные текстовые данные и формировать ответ в зависимости от них.
2. Возможность настраивать ответы

## Возможности в среде разработки

1. Запуск приложений на Java 11
2. Возможность слушать необходимый http порт любым приложением

## Прочие обстоятельства

1. Практическая работа на курсе “Java Professional”



# Инструмент, который бы не помешал

1. Возможность анализировать входные текстовые данные и формировать ответ в зависимости от них.
2. Возможность настраивать ответы

## Возможности в среде разработки

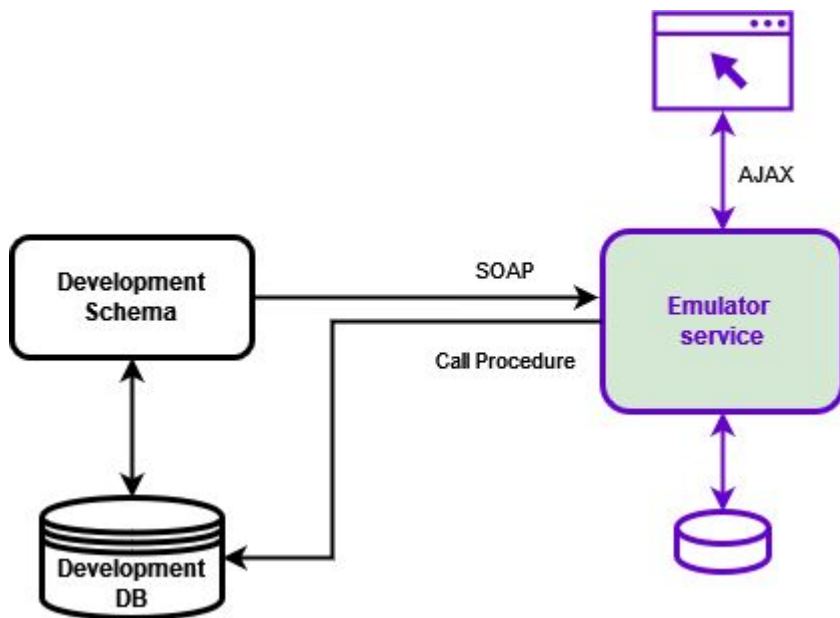
1. Запуск приложений на Java 11
2. Возможность слушать необходимый http порт любым приложением

## Прочие обстоятельства

1. Практическая работа на курсе “Java Professional”

Что делать? - Писать эмулятор!!!

# Эмулятор



Требования к эмулятору:

1. Эмулятор должен обрабатывать SOAP запросы анализируя данные
2. Формат входных и выходных бизнес данных должен настраиваться через конфигурационный файл
3. Для возможности “умного” формирования ответа сервис должен хранить предыдущие запросы
4. Эмулятор должен позволять “на ходу” настраивать своё поведение
5. Должен иметь Web-интерфейс
6. Данные эмулятора не должны попадать в базу основного приложения
7. Эмулятор должен уметь взаимодействовать напрямую с базой основного приложения

# Нефункциональные требования

1. Приложение на Spring boot 2.7
2. Spring Data JPA, работа с 2-мя различными БД одновременно (разные диалекты)
3. База самого приложения - безсерверная БД H2 или SQLite
4. База клиентского приложения - Oracle
5. Взаимодействие с Web-интерфейсом через REST API и Ajax

# Нефункциональные требования

1. Приложение на Spring boot 2.7
2. Spring Data JPA, работа с 2-мя различными БД одновременно (разные диалекты)
3. База самого приложения - безсерверная БД H2 или SQLite
4. База клиентского приложения - Oracle
5. Взаимодействие с Web-интерфейсом через REST API и Ajax

## “Вольности” допущенные в среде разработки

1. В качестве БД приложения на время разработки в основном использовал Postgre
2. В качестве SOAP клиента - Postman

# Нефункциональные требования

1. Приложение на Spring boot 2.7
2. Spring Data JPA, работа с 2-мя различными БД одновременно (разные диалекты)
3. База самого приложения - безсерверная БД H2 или SQLite
4. База клиентского приложения - Oracle
5. Взаимодействие с Web-интерфейсом через REST API и Ajax

## “Вольности” допущенные в среде разработки

1. В качестве БД приложения на время разработки в основном использовал Postgre
2. В качестве SOAP клиента - Postman

## Не реализовано

Полноценная работа с Oracle, в режиме эмуляции работы по асинхронному каналу

# Результат

POST SendTestMessage | POST SendTestMessage | POST SendTestMessage | GET http://cbr.ru/s/nev | GET ISNA Settings | + ... No Environment

Isna / SendTestMessage A03

POST http://localhost:8090/GovKgdService/SendMessage

Send

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL XML

Beautify

```
9      <serviceId>ISNA_BVU_BA_OPEN_CLOSE</serviceId>
10      <messageDate>2023-06-19T19:53:21+07:00</messageDate>
11      <sender/>
12      </requestInfo>
13      <requestData>
14      <data xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
15      xsi:type="xs:string">[CDATA[
16      :20:V386197160121274
17      :12:400
18      :77E:FORMS/A03/20230720/Увед. об изменении банковских счетов
19      /ACCOUNT/123456789/ 'KZ/KZ484324302398A00006/05/20230301/450509033484/20230301
20      -}}]]</data>
21      </requestData>
22      </request>
23      </SendTestMessage>
```

Body Cookies Headers (7) Test Results Status: 200 OK Time: 933 ms Size: 1.22 KB Save Response

Pretty Raw Preview Visualize XML

```
7      <messageId>725ff44c-8f69-4865-9c5a-4824c216a98f</messageId>
8      <correlationId>3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</correlationId>
9      <responseDate>2023-07-22T22:40:26.648+05:00</responseDate>
10     <status>
11       <code>OK</code>
12       <message>Message processed successfully</message>
13     </status>
14   </responseInfo>
15   <responseData>
16     <data xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
17     xsi:type="xs:string">[
18     :20:0010006500009024
19     :12:400
20     :77E:FORMS/A3C/20230722/Подтв.о получ.увед.об измен.номеров банк.счетов
21     /ACCOUNT/123456789/ 'KZ/KZ484324302398A00006/05/20230301/450509033484/99/Тестовая ошибка эмулятора!!!-</data>
22   </responseData>
23   </return>
```

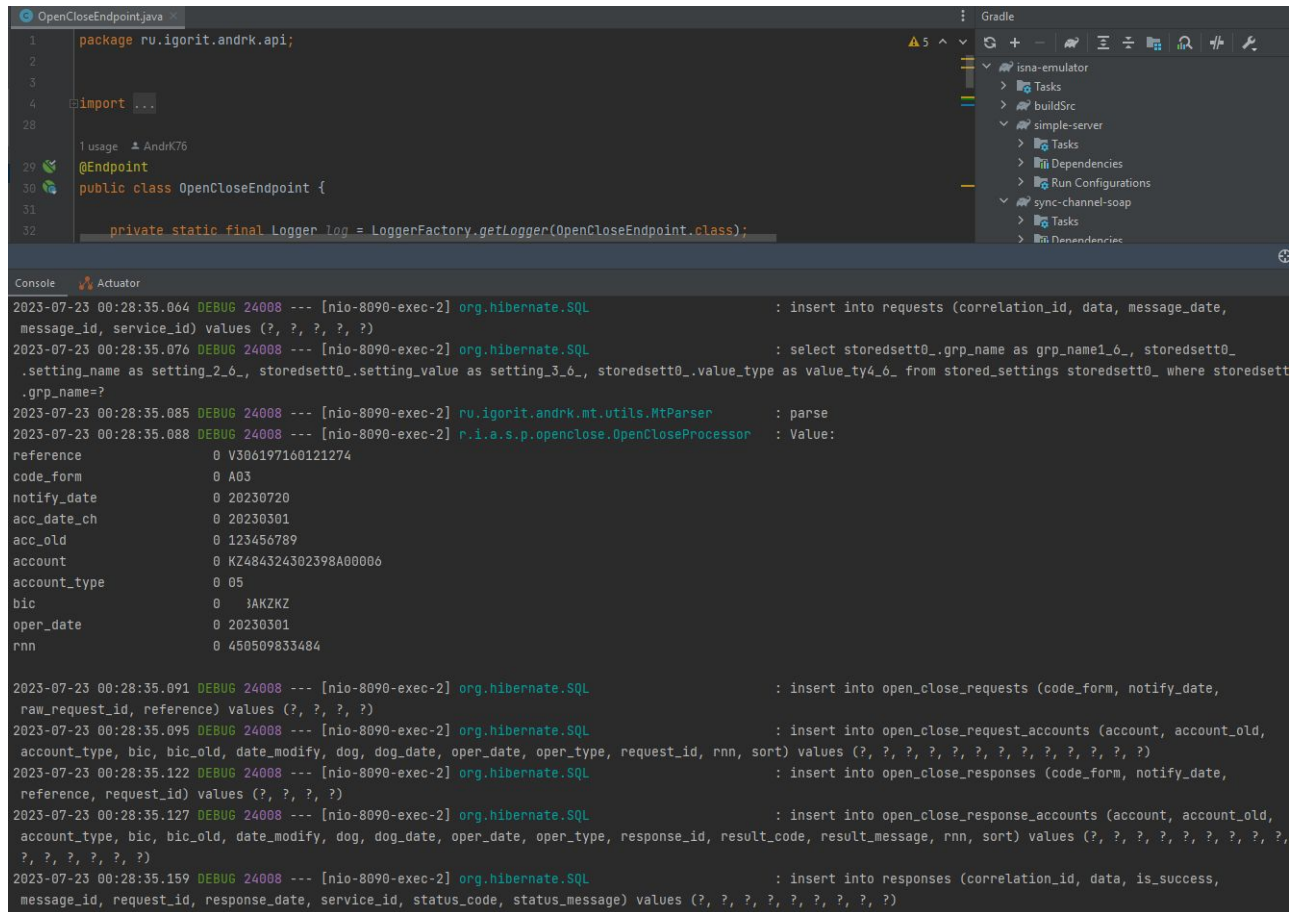
# Результат

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:8090/api/v1/opencloserequest?after=23&perPage=2`
- Method:** GET
- Body:** This request does not have a body.
- Status:** 200 OK
- Time:** 34 ms
- Size:** 1.07 KB
- Response Body (JSON):**

```
1  {
2    "content": [
3      {
4        "id": 22,
5        "rawRequestId": 23,
6        "messageId": "84c402af-45b5-474a-840f-f83eed4899ee",
7        "reference": "V306154735451685",
8        "codeForm": "A01",
9        "notifyDate": "2023-06-15T13:09:00",
10       "accounts": [],
11       "response": {
12         "id": 22,
13         "codeForm": "A1C",
14         "notifyDate": "2023-06-21T21:22:21",
15         "accounts": []
16       }
17     },
18     {
19       "id": 21,
20       "rawRequestId": 22,
21       "messageId": "3cd3b00a-502d-4e99-a44f-c4c1fa06ce41",
22       "reference": "V306154735451685",
23       "codeForm": "A01",
24       "notifyDate": "2023-06-15T13:09:00",
25       "accounts": [],
26       "response": {
27         "id": 21,
28         "codeForm": "A1C",
29         "notifyDate": "2023-06-21T21:22:21",
30         "accounts": []
31       }
32     }
33   ]
34 }
```

# Результат



The screenshot shows an IDE with a Java file named `OpenCloseEndpoint.java` and a console window displaying log output.

**Java Code:**

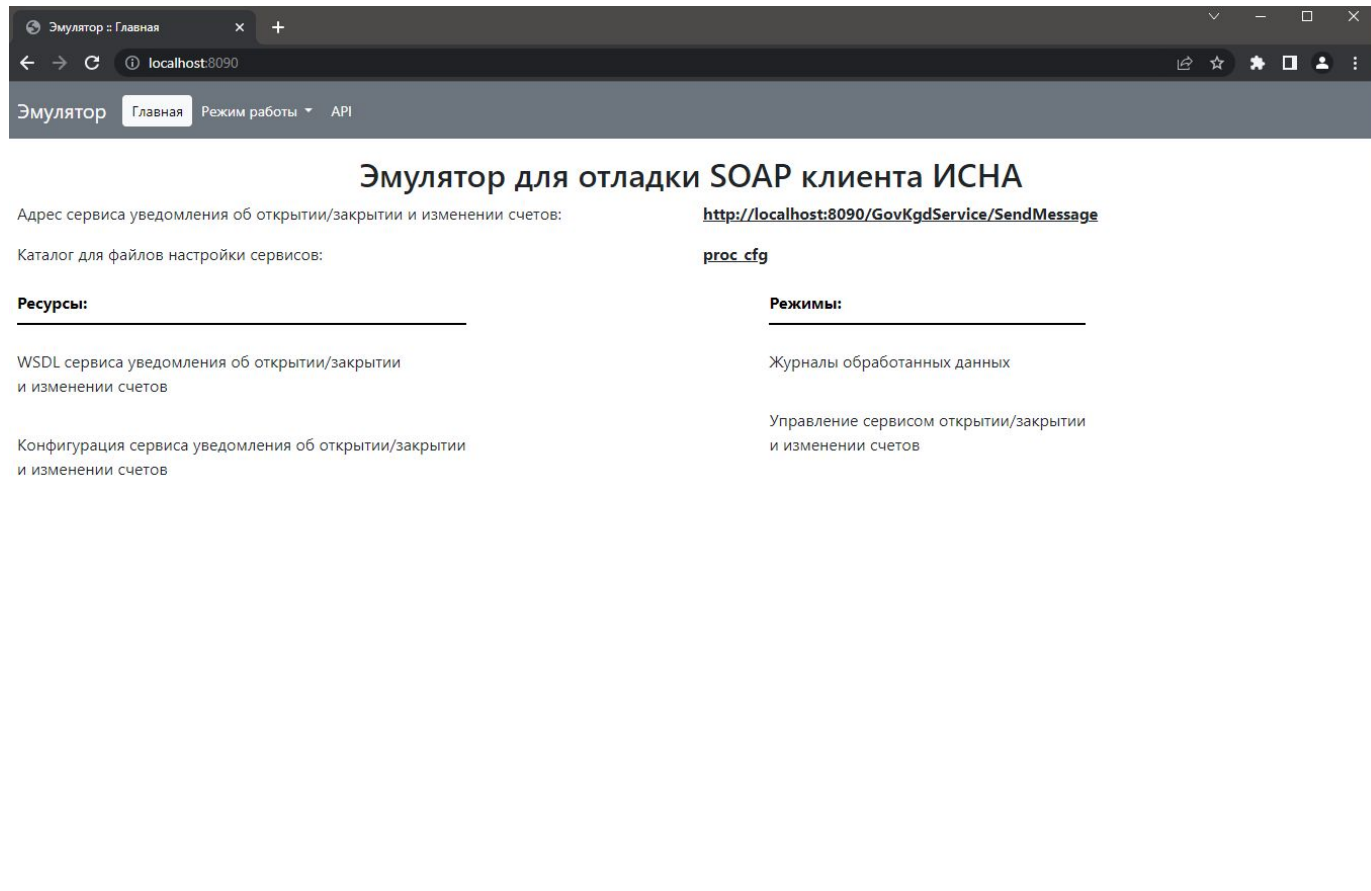
```
1 package ru.igorit.andrk.api;
2
3
4 import ...
28
1 usage  ▲ Andrk76
29 @Endpoint
30 public class OpenCloseEndpoint {
31
32     private static final Logger log = LoggerFactory.getLogger(OpenCloseEndpoint.class);
```

**Console Log:**

```
2023-07-23 00:28:35.064 DEBUG 24008 --- [nio-8090-exec-2] org.hibernate.SQL           : insert into requests (correlation_id, data, message_date,
message_id, service_id) values (?, ?, ?, ?, ?)
2023-07-23 00:28:35.076 DEBUG 24008 --- [nio-8090-exec-2] org.hibernate.SQL           : select storedset0_.grp_name as grp_name1_6_, storedset0_
.setting_name as setting_2_6_, storedset0_.setting_value as setting_3_6_, storedset0_.value_type as value_ty4_6_ from stored_settings storedset0_ where storedset
.grp_name=?
2023-07-23 00:28:35.085 DEBUG 24008 --- [nio-8090-exec-2] ru.igorit.andrk.mt.utils.MtParser      : parse
2023-07-23 00:28:35.088 DEBUG 24008 --- [nio-8090-exec-2] r.i.a.s.p.openclose.OpenCloseProcessor : Value:
reference           0 V306197160121274
code_form           0 A03
notify_date         0 20230720
acc_date_ch         0 20230301
acc_old             0 123456789
account             0 KZ484324302398A00006
account_type        0 05
bic                 0 3AKZKZ
oper_date           0 20230301
rnn                 0 450509833484
2023-07-23 00:28:35.091 DEBUG 24008 --- [nio-8090-exec-2] org.hibernate.SQL           : insert into open_close_requests (code_form, notify_date,
raw_request_id, reference) values (?, ?, ?, ?)
2023-07-23 00:28:35.095 DEBUG 24008 --- [nio-8090-exec-2] org.hibernate.SQL           : insert into open_close_request_accounts (account, account_old,
account_type, bic, bic_old, date_modify, dog, dog_date, oper_date, oper_type, request_id, rnn, sort) values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
2023-07-23 00:28:35.122 DEBUG 24008 --- [nio-8090-exec-2] org.hibernate.SQL           : insert into open_close_responses (code_form, notify_date,
reference, request_id) values (?, ?, ?, ?)
2023-07-23 00:28:35.127 DEBUG 24008 --- [nio-8090-exec-2] org.hibernate.SQL           : insert into open_close_response_accounts (account, account_old,
account_type, bic, bic_old, date_modify, dog, dog_date, oper_date, oper_type, response_id, result_code, result_message, rnn, sort) values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
2023-07-23 00:28:35.159 DEBUG 24008 --- [nio-8090-exec-2] org.hibernate.SQL           : insert into responses (correlation_id, data, is_success,
message_id, request_id, response_date, service_id, status_code, status_message) values (?, ?, ?, ?, ?, ?, ?, ?, ?)
```



# Результат



# Результат

Эмулятор

Главная

Режим работы

Все запросы

Открытие/закрытие

API

Статистика: Все запросы

Новее

Старше

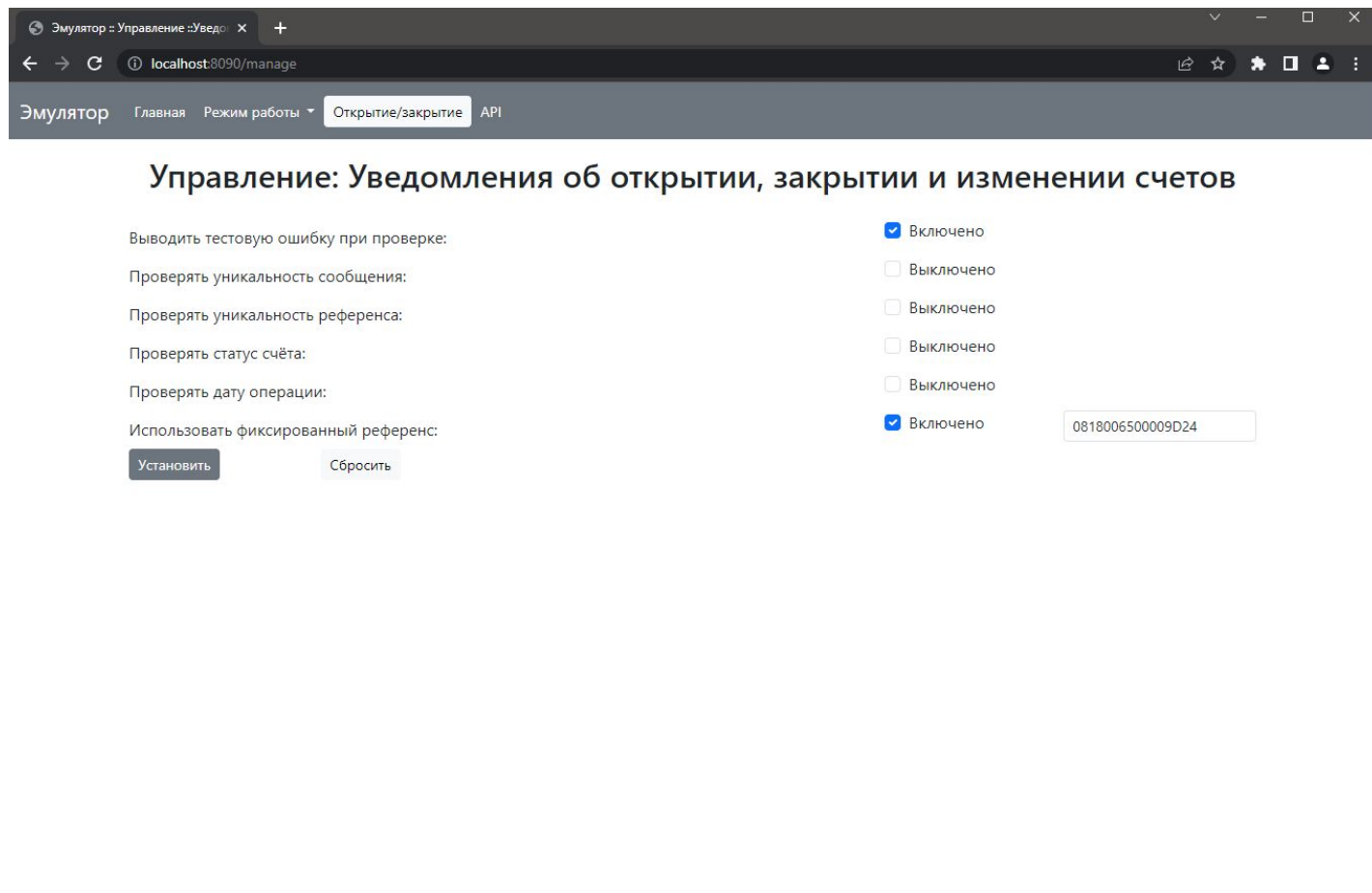
Показывать по: 10

Применить

Id сообщения	Дата	Сервис	Ответ	Дата ответа
<a href="#">3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</a>	2023-06-19 17:53:21 +05:00	ISNA_BVU_BA_OPEN_CLOSE	<a href="#">Message processed successfully (OK)</a>	2023-07-23 00:28:35 +05:00
Response: {4: :20:0818006500009024 :12:400 :77E:FORMS/A3C/20230723/Подтв.о получ.увед.об измен.номеров банк.счетов /ACCOUNT/123456789/ ZKZ/KZ484324302398A00006/05/20230301/450509833484/99/Тестовая ошибка эмулятора!!!-}				
<a href="#">3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</a>	2023-06-19 17:53:21 +05:00	ISNA_BVU_BA_OPEN_CLOSE	<a href="#">Message processed successfully (OK)</a>	2023-07-23 00:28:28 +05:00
<a href="#">3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</a>	2023-06-19 17:53:21 +05:00	ISNA_BVU_BA_OPEN_CLOSE	<a href="#">Error processing data (ERROR)</a>	2023-07-23 00:28:00 +05:00
<a href="#">3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</a>	2023-06-19 17:53:21 +05:00	ISNA_BVU_BA_OPEN_CLOSE	<a href="#">Error processing data (ERROR)</a>	2023-07-23 00:27:53 +05:00
<a href="#">3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</a>	2023-06-19 17:53:21 +05:00	ISNA_BVU_BA_OPEN_CLOSE	<a href="#">Error processing data (ERROR)</a>	2023-07-23 00:27:41 +05:00
<a href="#">3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</a>	2023-06-19 17:53:21 +05:00	ISNA_BVU_BA_OPEN_CLOSE	<a href="#">Message processed successfully (OK)</a>	2023-07-22 22:40:26 +05:00
<a href="#">3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</a>	2023-06-19 17:53:21 +05:00	ISNA_BVU_BA_OPEN_CLOSE	<a href="#">Message processed successfully (OK)</a>	2023-07-22 21:11:30 +05:00
<a href="#">3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</a>	2023-06-19 17:53:21 +05:00	ISNA_BVU_BA_OPEN_CLOSE	<a href="#">Message processed successfully (OK)</a>	2023-07-22 00:49:04 +05:00
<a href="#">3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</a>	2023-06-19 17:53:21 +05:00	ISNA_BVU_BA_OPEN_CLOSE	<a href="#">Message processed successfully (OK)</a>	2023-07-22 00:48:53 +05:00
<a href="#">3cd3b80a-502d-4e99-a44f-c4c1fa06ce42</a>	2023-06-19 17:53:21 +05:00	ISNA_BVU_BA_OPEN_CLOSE	<a href="#">Message processed successfully (OK)</a>	2023-07-22 00:48:38 +05:00

localhost:8090/list#

# Результат



# Результат

The screenshot displays an API emulator window titled "Эмулятор :: API definition". The address bar shows "localhost:8090/api". The interface includes a menu bar with "Главная" and "Режим работы", and a dropdown menu for "API".

The main area shows a request configuration for the endpoint `ISNA_BVU_BA_OPEN_CLOSE` with a `string (path)` type. Below this are "Execute" and "Clear" buttons.

The "Responses" section is active, displaying the following details:

- Curl:**

```
curl -X 'GET' \
'http://localhost:8090/api/v1/setting/ISNA_BVU_BA_OPEN_CLOSE' \
-H 'accept: */*'
```
- Request URL:**

```
http://localhost:8090/api/v1/setting/ISNA_BVU_BA_OPEN_CLOSE
```
- Server response:**
  - Code:** 200
  - Details:** Expanded to show the **Response body** as a JSON array:

```
[
  {
    "key": "CheckUniqueMessageId",
    "valueType": "java.lang.Boolean",
    "value": false
  },
  {
    "key": "CheckUniqueReference",
    "valueType": "java.lang.Boolean",
    "value": false
  },
  {
    "key": "FixReference",
    "valueType": "java.lang.String",
    "value": "0818006500009024"
  }
]
```

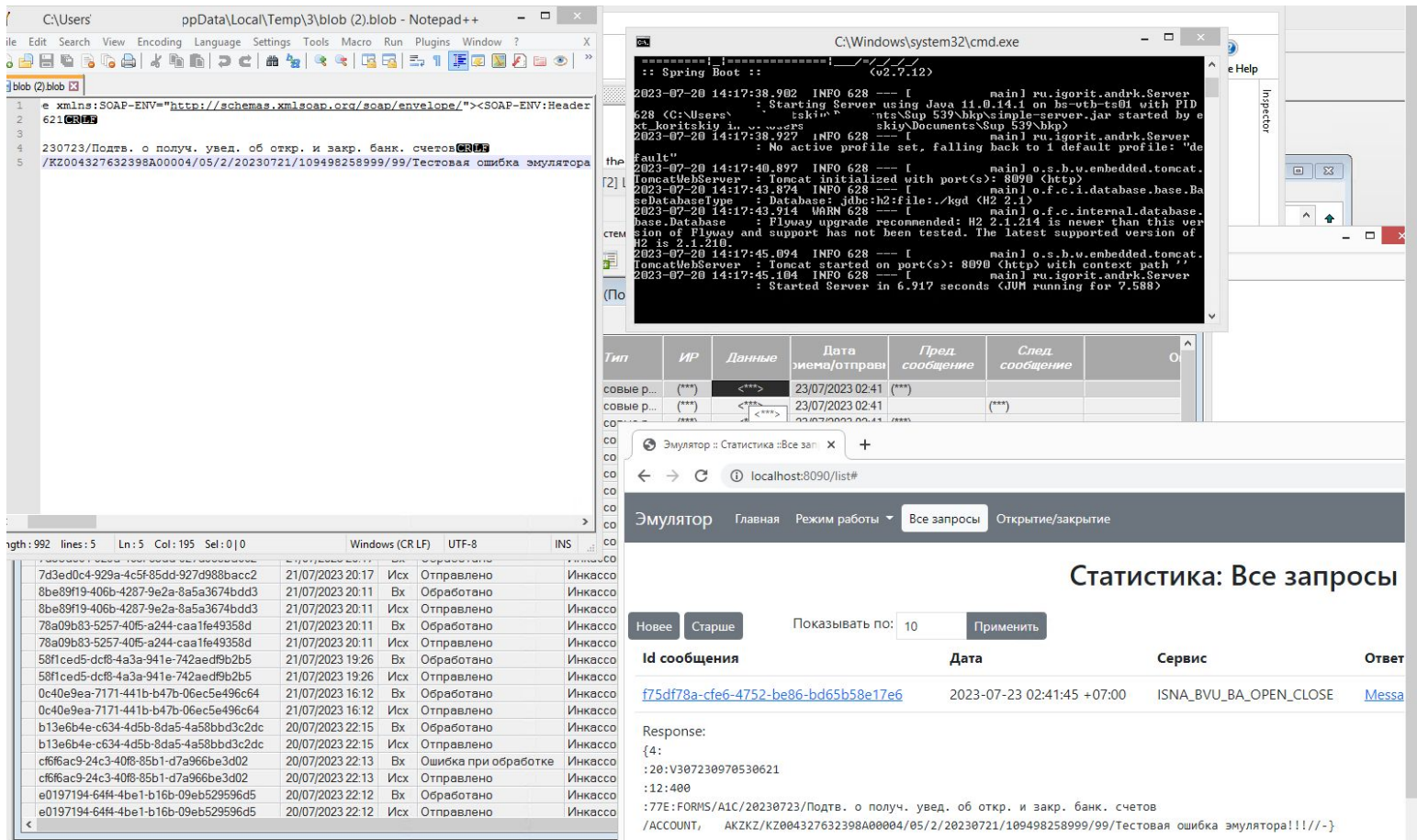
# Результат

The image shows a web browser window at localhost:8090 displaying the 'Эмулятор для отладки SOAP клиента ИСНА' (ISNA SOAP Client Emulator). The browser interface includes a navigation bar with 'Главная', 'Режим работы', and 'API' tabs. The main content area has a title 'Эмулятор для отладки SOAP клиента ИСНА' and a description: 'Адрес сервиса уведомления об открытии/закрытии счетов' and 'Каталог для файлов настройки сервисов:'. Below this, there is a section 'Ресурсы:' with two links: 'WSDL сервиса уведомления об открытии/закрытии счетов' and 'Конфигурация сервиса уведомления об открытии/закрытии счетов'.

Overlaid on the browser is a Visual Studio Code window showing the file 'ISNA\_BVU\_BA\_OPEN\_CLOSE (3).cfg'. The code is an XML Schema Definition (XSD) for a SOAP message. It starts with a root element 'input' containing a 'CDATA' section. The 'CDATA' section contains a series of elements: 'SUBJECT', 'ACCOUNT', 'ID', 'SUBJECT', 'ACCOUNT', 'ACCOUNT\_CHANGE', and 'input'. The 'input' element is a complex type with a 'format' attribute. The 'format' attribute is defined in the 'Формат входящего документа' section, which specifies the structure of the input data. The 'Формат строки' section defines the format of the string, and the 'Формат итема' section defines the format of the item. The 'Имя: обязательность: фиксированная\_длина: тип: длина: формат' section defines the format of the item. The 'Тип итема' section defines the type of the item, and the 'Имя: обязательность: фиксированная\_длина: тип: длина: формат' section defines the format of the item.

```
1 <?xml version="1.0"?>
2 <data>
3   <!--
4     Формат входящего документа
5     Код уровня~информация специфичная для уровня
6     Уровень 1: Список секций
7     Формат: Код секции~маска поиска~количество повторов
8     Уровень 2: Секции для предразбора
9     Уровень 3: Секции для окончательного предразбора
10    Формат: Код секции~Разделитель итемов~Формат строки
11
12    Формат строки:
13    То что не в {} - должно быть обязательно, в {} описаны итемы
14
15    Формат итема:
16    Имя: обязательность: фиксированная_длина: тип: длина: формат
17    Тип итема:
18    x - строка
19    d - дата
20    i - целое число
21  -->
22  <input>
23    <![CDATA[1~ID~1~:20~1
24    1~SUBJECT~2~:77E:FORMS/~1
25    1~ACCOUNT~3~:ACCOUNT/~N
26    2~ID~/~:20:{reference:1:1:x:16}
27    2~SUBJECT~/~:77E:FORMS/{code_form:1:1:x:3}/{notify_date:1:1:d:8:yyyyMMdd}
28    3~ACCOUNT~/~:ACCOUNT/{bic:1:0:x:11}/{account:1:0:x:20}/{account_type:1:0:x:5}/{oper_type:1:0:i:1}/{oper_date:1:0:d:8:yyyyMMdd}/{rnn:1:
29    3~ACCOUNT_CHANGE~/~:ACCOUNT/{acc_old:0:0:x:20}/{bic:1:0:x:11}/{account:1:0:x:20}/{account_type:1:0:x:5}/{oper_date:1:1:d:8:yyyyMMdd}/{
30  </input>
31  <!--
32    11x{incass_detbic}/20x{incass_detacc}/2x{incass_detacctype}/1x{incass_detotype}/80{incass_detacccdate:yyyymmdd}/12x{incass_detrnn}
33    11x{incass_detbic}/20x{incass_detacc}/2n{incass_detacctype}/80{incass_detacccdate:yyyymmdd}/12x{incass_detrnn}/[11x{incass_detbic}n
```

## Результат



Live...