ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ «АВАНГАРД. ЭЛЕКТРОННЫЙ ДОКУМЕНТООБОРОТ»

РУКОВОДСТВО СИСТЕМНОГО ПРОГРАММИСТА

Листов 69

Москва
2023

АННОТАЦИЯ

Документ содержит общие сведения о назначении и архитектуре программного обеспечения «Авангард. Электронный документооборот» (далее – Система), требования к аппаратному и программному обеспечению Системы, к квалификации персонала, сопровождающего Систему. Подробно описаны процессы жизненного цикла программного обеспечения на стадиях внедрения и сопровождения:

– первоначальная установка и настройка программного обеспечения Системы;

– поддержание работоспособности программного обеспечения Системы, в том числе восстановление Системы после сбоев;

– обновление программного обеспечения Системы.

Документ не содержит описания процесса вывода программного обеспечения Системы из эксплуатации.

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

Сокращенное наименование	Полное наименование
БД	База данных
ОС	Операционная система
Система	Программное обеспечение «Авангард. Электронный документооборот»
СУБД	Система управления базами данных

1. Общие сведения о системе

1.1. Назначение Системы

Программное обеспечение «Авангард. Электронный документооборот» предназначено для обеспечения процессов формирования, подписания и хранения документов в электронной форме, а также автоматизации процесса внутреннего согласования и утверждения электронных документов.

1.2. Состав Системы

Система представляет собой web-приложение и включает в свой состав следующий набор компонент:

- системные компоненты:
  - сервис «Администрирование и контроль доступа (Хаб)» (далее также – Хаб);
  - сервис «Файловое хранилище»;
  - сервис «Справочники»;
  - сервис «Уведомления»;
  - сервис «Оболочка»;
- прикладные компоненты:
  - сервис «Хранение и оборот документов»;
  - сервис «ОШС» (организационно-штатная структура).

Прикладные компоненты состоят из:

- Базы данных;
- Сервера приложения;
- Клиентского приложения (фрагментарно загружается на машину пользователя).

В рамках Системы используются вспомогательные сторонние сервисы с открытым исходных кодом:

- Программный брокер сообщений (RabbitMQ);
- Платформа для обнаружения и анализа контента (Apache Tika).

1.3. Требования к техническому и программному обеспечению

1.3.1. Требования к программному обеспечению

Состав программного обеспечения пользовательской рабочей станции:

- браузер Google Chrome, Microsoft Edge, Яндекс Браузер последней или предпоследней версии;

– наличие актуальной версии установленных драйверов оборудования.

Требования к программному обеспечению тестового и продуктивного серверов:

– серверная операционная система семейства Linux;
– СУБД PostgresPro либо PostgreSql версии не ниже 11;
– СУБД MongoDB 6 или MinIO;
– прокси Nginx не ниже 1.22;
– программная платформа .NET sdk 6.0.

1.3.2. Требования к техническому обеспечению

К аппаратному обеспечению серверной части, предназначенной для обеспечения функционирования Системы (не более 100 одновременных подключений пользователей), предъявляются следующие минимальные требования:

– Процессоры:
  - количество не менее 2;
  - архитектура процессора x86-64;
  - ядер не менее 8;
  - потоков не менее 16;
  - тактовая частота в режиме повышенной нагрузки не менее 3,3 ГГц;
  - кэш не менее 20 Мб;
  - поддержка памяти ECC.
– Оперативная память:
  - объем не менее 64 Гб;
  - тип DDR3/DDR4/DDR5 с функцией коррекции ошибок.
– Сетевой интерфейс:
  - не менее 1 порта 100 Мб/с.
– Дисковая подсистема:
  - аппаратный RAID;
  - интерфейс SAS не менее 6 Гб/сек;
  - HDD с буфером обмена не менее 128 Мб либо SSD.
– Коммуникационная среда должна обеспечивать информационное взаимодействие между компонентами Системы в соответствии с транспортным протоколом TCP/IP.

К аппаратному обеспечению рабочей станции пользователя, предназначенной для обеспечения доступа к функционалу Системы, предъявляются следующие требования:

- Процессоры:
  - количество не менее 1;
  - архитектура процессора x86-64;
  - ядер не менее 2;

  Допустимо использование следующих видов процессоров:
  - настольные процессоры Intel и AMD, вышедшие на рынок не ранее 2014 года;
  - мобильные процессоры Intel и AMD, вышедшие на рынок не ранее 2016 года, кроме линейки процессоров Intel Atom;
  - процессоры Apple линейки M1, M2 и более новые.
- Оперативная память:
  - объем не менее 4 Гб (рекомендуется 8 Гб);
  - тип DDR3/DDR4/DDR5.
- Сетевой интерфейс:
  - не менее 1 порта 100Мб/с
  - (доступ к сервисам системы со скоростью не ниже 8 Мбит/с (для быстрой загрузки приложения рекомендуется 25 Мбит/с и выше).
- Дисковая подсистема:
  - HDD с буфером обмена не менее 64 Мб либо SSD.
- Графический режим монитора:
  - 1366x768 и выше (рекомендуется 1920x1080).
- Клавиатура, мышь.

1.4. Требования к квалификации персонала, обеспечивающего эксплуатацию Системы

Персонал, выполняющий функции технического сопровождения Системы, должен обладать экспертными навыками:

- обеспечения функционирования серверов и рабочих станций в среде ОС семейства Linux;
- установки и настройки программного обеспечения, приведённого в п. 2.

2. Установка и настройка программного обеспечения

2.1. Предварительные шаги установки дистрибутива

2.1.1. Установка Postgresql

Установка СУБД на примере дистрибутива Postgresql-11 из базового репозитория AstraLinux 1.7:

Обновить информацию о пакетах и установить СУБД:

```
apt update
apt install postgresql-11
```

2.1.2. Установка Dotnet

Необходимо установить среду исполнение ***dotnet***.

1) Скачать дистрибутив по ссылке:

https://dotnet.microsoft.com/en-us/download/dotnet/thank-you/sdk-6.0.403-linux-x64-binaries

2) Скопировать полученный архив в пустой каталог на сервер с помощью SCP-клиента (для windows можно использовать WinSCP)

3) Распаковать архив командой:

```
tar xvf dotnet-sdk-6.0.403-linux-x64.tar.gz
```

Если планируется, что web-сервер Kestrel будет использовать https в качестве ApplicationUrl приложений, то следует выполнить команду для установки сертификатов ***dotnet***:

```
dotnet dev-certs https --clean
dotnet dev-certs https –t
```

В противном случае, терминирование должно происходит на ***nginx***, а любое общение между микросервисами должно происходить с использованием внешнего адреса.

2.1.3. Установка Rabbit MQ

Установка проводится следующим образом:

```
apt update
apt install rabbitmq-server
```

2.1.4. Установка nginx

1) Необходимо подключить Extended-репозиторий Astralinux 1.7. Для этого внесем изменения в файл ***/etc/apt/sources.list*** командой:

```
nano /etc/apt/sources.list
```

Добавим текст с репозиториями:

Основной репозиторий
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-main/ 1.7_x86-64 main contrib non-free

Оперативные обновления основного репозитория
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-update/ 1.7_x86-64 main contrib non-free

Базовый репозиторий
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-base/ 1.7_x86-64 main contrib non-free

Расширенный репозиторий
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-extended/ 1.7_x86-64 main contrib non-free

2) Обновим информацию о пакетах и установим ***nginx***:

apt update
apt install nginx

Подготовить ключ (***/usr/share/crt/private.key***) и сертификат (***/usr/share/crt/cert.crt***) который будут использоваться для SSL шифрования (***https***).

2.1.5. Установка Redis

Установка проводится следующим образом:

apt update
apt install redis-server

2.2. Установка и развертывание сервиса «Администрирование и контроль доступа (Хаб)»

2.2.1. Подготовка БД

1) Подключиться по ***ssh*** к серверу, на котором развернута СУБД;
2) Выполнить команду ***psql***;
3) Выполнить команду create database app.hub;
4) Выполнить команду create database app.logs.

2.2.2. Развёртывание БД логов

Выполнить скрипт на ***app.logs***

```
create extension "uuid-ossp";

create schema logs;

create table logs.auth
(
   id         uuid    not null,
   login      text,
   client     text,
   is_success boolean not null,
   remote_ip  text,
   local_ip   text,
   date_time  timestamp with time zone default now()
);

create index auth_date_created
   on logs.auth (date_time);

create table logs.access
(
   id              uuid                  default uuid_generate_v4() not null,
   client          text,
   resource        text,
   login           text,
   claim           text,
   role_id         uuid                                    not null,
   organization_id uuid                                    not null,
   is_success      boolean                                 not null,
   description     text,
   date_time       timestamp with time zone default now()
);

create index access_login_indexed
   on logs.access (login);

create index access_date_created
   on logs.access (date_time);

create function logs.add_auth_event(data text) returns void
```

```
    language plpgsql
as
$$
DECLARE
    _json_data json = data::json;
    _user_name text = _json_data ->> 'UserName';
    _client_id text = _json_data ->> 'ClientId';
    _timestamp timestamp = (_json_data ->> 'TimeStamp')::timestamp;
    _is_success boolean = (_json_data ->> 'IsSuccess')::boolean;
    _local_ip text = _json_data ->> 'LocalIpAddress';
    _remote_ip text = _json_data ->> 'RemoteIpAddress';
BEGIN

    insert into logs.auth(id, login, client, is_success, remote_ip, local_ip,
date_time)
    select uuid_generate_v4(), _user_name, _client_id, _is_success,
_remote_ip, _local_ip, _timestamp;
    end
$$;

create function logs.add_access_event(data text) returns void
    language plpgsql
as
$$
DECLARE
    _json_data json = data::json;
    _user_name text = _json_data ->> 'UserName';
    _claim text = _json_data ->> 'Claim';
    _role_id uuid = (_json_data ->> 'RoleId')::uuid;
    _org_id uuid = (_json_data ->> 'OrganizationId')::uuid;
    _is_success boolean = (_json_data ->> 'IsSuccess')::boolean;
    _timestamp timestamp = (_json_data ->> 'TimeStamp')::timestamp with
time zone;
    _description text = _json_data ->> 'Description';
    _client text = _json_data ->> 'Client';
    _resource text = _json_data ->> 'Resource';
BEGIN

    insert into logs.access(id, client, resource, login, claim, role_id,
organization_id, is_success, description, date_time)
```

```
        select uuid_generate_v4(), _client, _resource, _user_name, _claim,
_role_id, _org_id, _is_success, _description, _timestamp;
    end
$$;

    create function logs.get_auth_events(take integer DEFAULT 50, skip integer
DEFAULT 0, sort text DEFAULT 'date_time'::text, dir text DEFAULT 'desc'::text,
where_clause text DEFAULT NULL::text) returns SETOF refcursor
    language plpgsql
as
$$
DECLARE
    _sql1           text;
    _sql2           text;
    _refcursor      refcursor = 'data_cursor';
    _refcursor_total refcursor = 'total_cursor';
BEGIN
    _sql1 := 'SELECT login, client, is_success, date_time FROM logs.auth '
        || where_clause;

    _sql2 := 'SELECT COUNT(1) AS total FROM (' || _sql1 || ') as sub';

    _sql1 := _sql1 || ' ORDER BY ' || sort || ' ' || dir
            || ' LIMIT ' || take
            || ' OFFSET ' || skip;

    OPEN _refcursor FOR
        EXECUTE _sql1;
    RETURN NEXT _refcursor;


    OPEN _refcursor_total FOR
        EXECUTE _sql2;
    RETURN NEXT _refcursor_total;

END;

$$;
```

```
create function logs.get_access_events(take integer DEFAULT 50, skip
integer DEFAULT 0, sort text DEFAULT 'date_time'::text, dir text DEFAULT
'desc'::text, where_clause text DEFAULT NULL::text) returns SETOF refcursor
    language plpgsql
as
$$
DECLARE
    _sql1 text;
    _sql2 text;
    _data_cursor  refcursor = 'data_cursor';
    _total_cursor refcursor = 'total_cursor';
BEGIN

    _sql1 := 'SELECT client, resource, login, claim, role_id, organization_id,
is_success, description, date_time
    FROM logs.access ' || where_clause;

    _sql2 := 'SELECT COUNT(1) AS total FROM (' || _sql1 || ') as sub';

    _sql1 := _sql1 || ' ORDER BY ' || sort || ' ' || dir
            || ' LIMIT ' || take
            || ' OFFSET ' || skip;

    OPEN _data_cursor FOR
      EXECUTE _sql1;
    RETURN NEXT _data_cursor;

    OPEN _total_cursor FOR
      EXECUTE _sql2;
    RETURN NEXT _total_cursor;
end;
$$;
```

2.2.3. Настройка бэкенд-приложения

1) Скопировать дистрибутив сервиса в директорию сервера (например, ***/usr/share/hosting/hub***);

2) Создать пользователя, под которым будет запускаться приложение. Пользователь должен иметь полный доступ к директории с приложением и права открывать сокеты;

3) Перейти в директорию дистрибутива ***/clients/default***;

4) Выполнить настройку ***appsettings.json***.

```
{
 "ConnectionStrings": {
  "DefaultConnection": "Server=localhost; Database=app.hub; User
ID=postgres; Password=postgres",
  "LogsConnection": "Server=localhost; Port=5433; Database=app.logs; User
ID=postgres; Password=postgres"
 },
 "ApplicationUrls": [
  "https://localhost:1900"
 ],
 "CorsOrigins": [
  "^http[s]?:\\/\\/x\\.x\\.x\\.x(:\\d{1,6})?$",
  "^http[s]?:\\/\\/localhost(:\\d{1,6})?$"
 ],
 "HostedServices": {
  "Items": [
   {
    "Key": "OrganizationReplication",
    "Enabled": false,
    "Interval": "00:05:00"
   },
   {
    "key": "GridStateCaching",
    "Enabled": false
   },
   {
    "key": "PersonalFileImport",
    "Enabled": false,
    "Interval": "00:05:00"
   }
  ]
 },
 "Logging": {
  "IncludeScopes": false,
  "LogLevel": {
   "Default": "Warning",
   "System": "Warning",
   "Microsoft": "Warning"
  }
```

```
  },
  "Customization": {
   "GridState": {
    "FallbackOnNoCache": false
   }
  },
  "Navigation": {
   "Origins": {
    "Auth": "https://x.x.x.x" //Внешний адрес сервиса «Авторизация»
   }
  },
  "Workspace": {
   "OrgMode": "multi",
   "RoleMode": "multi"
  },
 "EventBus": {
   "Enabled": true,
   "Broker": "app.workspace",
   "RetryCount": 10,
   "QueueName": "app.hub",
   "CommandQueueName":"app.hub",
   "ExchangeType": "direct",
   "BusAccess": {
    "Host": "x.x.x.x", /ip-адрес RabbitMQ
    "UserName": "user", //логин
    "Password": "password", //пароль
    "RetryCount": 10
   }
  },
  "Smtp": {
   "Host": "", //имя хоста почтового сервера
   "Port": 465, //порт
   "IsSsl": false, //использоваться ли ssl при подключении
   "Email": "", //адрес с которого отправлять сообщение
   "UserName": "", //логин подключения к почте
   "Password": "", /пароль подключения к почте
   "CheckCertificateRevocation": false //проверять ли ssl сертификат на
актуальность
  },
  "Authentication": {
   "Authority": "https://x.x.x.x"  , //Внешний адрес сервиса Хаб
```

```
        "ApiName": "identityServer", //идентификатор ресурса (для запросов на API используя JWT из SPA-приложения в составе сервиса Хаб)
        "ApiSecret": "auth", //Секретное слово
        "ClaimType": "uri://schemas.quarta.su/permission-claim-type",
        "Password": {     //Настройка политик безопасности (в отношении новых паролей), уже созданным учеткам будет разрешено войти с их действующими паролями
          "RequireDigit": false,
          "RequiredLength": 1,
          "RequireNonAlphanumeric": false,
          "RequireUppercase": false,
          "RequireLowercase": false,
          "RequiredUniqueChars": 1
        },
        "AccountBlocking": {
          "MaxLoginAttempts": 99       // Максимальное количество неудачных попыток входа (правильный логин, неправильный пароль)
        },
        "Identity": { //Настройка SPA-приложения. Указанные настройки заливаются в БД при запуске приложения
          "ClientId": "admin",
          "ClientName": "admin",
          "Host": "https://x.x.x.x", //публичный хост сервиса «Авторизация»
          //Адреса редиректов куда разрешено вернуться после удачной аутентификации
          //Следует менять только Хост.
          "CallbackUrl": [
            "https://x.x.x.x/login-callback",
            "https://x.x.x.x/silent-callback"
          ],
          //Адреса на которые разрешено вернуться после разлогинивания
          "PostLogoutUrl": "https://x.x.x.x/login",
          "Scope": "openid profile identityServer"
        }
      },
      "FileStorage": {
        "UseLocalStorage": true
      }
    }
```

2.2.4. Настройка systemd-сервиса

1) Создать systemd-файл (например, ***/etc/systemd/system/hub.service***);

2) Ввести настройки запуска сервиса ***dotnet*** приложением с указанием параметров. Входной точкой приложения является ***Quarta.Auth.Web.dll***.

3) Выполнить команду ***systemctl daemon-reload***, чтобы ввести сервис в действие.

4) Проверить состояние сервиса командой ***systemctl status hub***;

Пример настроек ***hub.service***

```
[Unit]
Description=[Hub] Авторизация
[Service]
WorkingDirectory=/usr/share/hosting/hub
ExecStart=/usr/bin/dotnet /usr/share/hosting/hub/Quarta.Auth.Web.dll
Restart=always
RestartSec=10
SyslogIdentifier=dotnet-bi
User=%ИМЯ СИСТЕМНОГО ПОЛЬЗОВАТЕЛЯ%
Environment=DOTNET_PRINT_TELEMETRY_MESSAGE=false
Environment=DOTNET_CLI_TELEMETRY_OPTOUT=false
Environment=ASPNETCORE_ENVIRONMENT=Production
[Install]
WantedBy=multi-user.target
```

2.2.5. Настройка nginx

1) Перейти в директорию ***/etc/nginx/conf.d*** (или аналогичную директорию для размещения динамический модулей ***nginx***);

2) Создать файл app-hub-service.conf;

3) Внести настройки proxy_pass между внешним адресом сервера и внутренним;

4) Проверить корректность настроек командой ***nginx –t***;

5) Выполнить команду ***nginx reload*** если проверка прошла успешно;

Пример настроек ***app-hub-service.conf***

```
server {
   listen 1900 ssl;
```

```
        location / {
            proxy_pass https://127.0.0.1:1900;
        }
    }
```

Адрес proxy_pass должен соответствовать значению поля ApplicationUrl указанного в ***appSettings.json***.

2.2.6. Настройка SPA-приложения

1) Перейти в директорию дистрибутива wwwroot/app/assets/configurations/clients/default;
2) Открыть файл configuration.json;
3) Внести настройки SPA-приложения.

Пример настроек ***configuration.json***

```
{
  "serverUrl": "https://x.x.x.x", //публичный адрес сервиса Хаб
  "authentication": {
    "authority": "https://x.x.x.x", //публичный адрес сервиса Хаб
    "login_uri": "https://x.x.x.x/login",
    "redirect_uri": "https://x.x.x.x./login-callback",
    "silent_redirect_uri": "https://x.x.x.x/silent-callback",
    "post_logout_redirect_uri": "https://x.x.x.x/login",
    "client_id": "admin",
    "response_type": "id_token token",
    "scope": "openid profile identityServer"
  },
  "navigation": {
    "api": "https://x.x.x.x/api/navigation", //публичный адрес сервиса Хаб
    "domain": "Auth"
  },
  "filestorage": {
    "fileStorageUrl": "https://x.x.x.x/api/file-storage/", //публичный адрес сервиса Хаб
    "maxAllowedFileSize": 1073741824
  }
}
```

Настройки секции authentication должны соответствовать значениям из конфига бэкенд-приложения ***Authentication:Identity***.

2.3. Установка и развертывание сервиса «Файловое хранилище»

2.3.1. Подготовка БД

1) Скопировать файлы дистрибутива mongodb mongodb-org-server_6.0.3_amd64.deb и mongodb-org-mongos_6.0.3_amd64.deb в директорию на сервере.

2) Установить MongoDB, выполнив команды:

```
sudo dpkg -i mongodb-org-server_6.0.3_amd64.deb
        sudo dpkg -i mongodb-org-mongos_6.0.3_amd64.deb
```

3) Запустить службу MongoDB, выполнив команды:

```
systemctl start mongod
systemctl enable mongod
```

4) Убедиться, что MongoDB слушает свой порт, выполнив команду:

```
netstat -tunlp | grep -i mongo
```

5) Создать базу и collection в ней (пока что пустой), выполнив команды:

```
mongosh
use filestorage
db.files.insertOne( { x: 1 } )
```

В результате возможен запуск на ***localhost*** без пароля.

2.3.2. Настройка бэкенд-приложения

1) Скопировать дистрибутив сервиса в директорию сервера (например, ***/usr/share/hosting/fs***)

2) Создать пользователя, под которым будет запускаться приложение. Пользователь должен иметь полный доступ к директории с приложением и права открывать сокеты;

3) Выполнить настройку ***appsettings.json***.

```
{
  "ApplicationUrls": [
    "https://localhost:5040"
  ],
  "MongoDB": {
```

```
    "ConnectionString": "mongodb://x.x.x.x.:27017", //адрес где размещена
СУБД «MongoDB”
    "Catalog": "production",
    "DefaultCollection": "files"
   },
   "Logging": {
    "LogLevel": {
     "Default": "Information",
     "Microsoft": "Warning",
     "Microsoft.Hosting.Lifetime": "Information"
    }
   },
   "AllowedHosts": "*"
  }
```

2.3.3. Настройка systemd-сервиса

1) Создать systemd-файл (например, ***/etc/systemd/system/fs.service***);

2) Ввести настройки запуска сервиса ***dotnet*** приложением с указанием параметров. Входной точкой приложения является ***Quarta.FileStorage.NetCore.Web.dll***.

3) Выполнить команду ***systemctl daemon-reload***, чтобы ввести сервис в действие.

4) Проверить состояние сервиса командой ***systemctl status fs***.

Пример настроек ***fs.service***

```
[Unit]
Description=[fs] Файловое хранилище
[Service]
WorkingDirectory=/usr/share/hosting/fs
ExecStart=/usr/bin/dotnet
/usr/share/hosting/fs/Quarta.FileStorage.NetCore.Web.dll
Restart=always
RestartSec=10
SyslogIdentifier=dotnet-bi
User=%ИМЯ СИСТЕМНОГО ПОЛЬЗОВАТЕЛЯ%
Environment=DOTNET_PRINT_TELEMETRY_MESSAGE=false
Environment=DOTNET_CLI_TELEMETRY_OPTOUT=false
Environment=ASPNETCORE_ENVIRONMENT=Production
[Install]
```

```
WantedBy=multi-user.target
```

2.3.4. Настройка nginx

1) Перейти в директорию ***/etc/nginx/conf.d*** (или аналогичную директорию для размещения динамический модулей ***nginx***);
2) Создать файл app-fs-service.conf;
3) Внести настройки proxy_pass между внешним адресом сервера и внутренним;
4) Проверить корректность настроек командой ***nginx –t***;
5) Выполнить команду ***nginx reload***, если проверка прошла успешно.

Пример настроек ***app-fs-service.conf***

```
server {
                listen       1040 ssl;
                location / {
                    proxy_pass https://127.0.0.1:5040;
                }
 }
```

Адрес proxy_pass должен соответствовать значению поля ApplicationUrl указанного в ***appSettings.json***.

2.4. Установка и развертывание сервиса «Справочники»

2.4.1. Подготовка БД

1) Подключиться по ***ssh*** к серверу, на котором развернута СУБД;
2) Выполнить команду ***psql***.
3) Выполнить команду create database app.classifier.

2.4.2. Настройка бэкенд-приложения

1) Скопировать дистрибутив сервиса в директорию сервера (например, ***/usr/share/hosting/classifier)***.
2) Создать пользователя, под которым будет запускаться приложение. Пользователь должен иметь полный доступ к директории с приложением и права открывать сокеты;
3) Перейти в директорию дистрибутива ***/clients/default***;
4) Выполнить настройку ***appsettings.json***.

```
{
  "ApplicationUrls": [ "https://localhost:1300" ],
  "ConnectionStrings": {
    "DefaultConnection": "Server=localhost; Database=app.classifier; User ID=postgres; Password=postgres"
  },
 "AuthServer": {
    "Url": "https://x.x.x.x", //хост приложение сервиса Хаб. Может указываться внешний или внутренний адрес
    "ApiName": "", //идентификатор данного сервиса (режим «Ресурсы» сервиса Хаб)
    "ApiSecret": "", //пароль данного сервиса (режим «Ресурсы» сервиса Хаб, код авторизации)
    "ClaimType": "uri://schemas.quarta.su/permission-claim-type",
    "AccessEventEndpointName": "access_event_endpoint"
  },
  "EventBus": {
    "Enabled": true,
    "Broker": "app.workspace",
    "RetryCount": 10,
    "QueueName": "app.classifier”,
    "CommandQueueName":"app.classifier",
    "ExchangeType": "direct",
    "BusAccess": {
      "Host": "x.x.x.x", //ip-адрес RabbitMQ
      "UserName": "user", //логин
      "Password": "password", //пароль
      "RetryCount": 10
    }
  },
  "Logging": {
    "LogLevel": {
      "Default": "Warning",
      "System": "Warning",
      "Microsoft": "Warning"
    }
  },
  "FileStorage": {
    "Url": http://x.x.x.x//адрес файлового хранилища.
  },
```

```
  "CorsOrigins": [
    "^http[s]?:\\/\\/x\\.x\\.x\\.x(:\\d{1,6})?$",
    "^http[s]?:\\/\\/localhost(:\\d{1,6})?$"
  ]
}
```

2.4.3. Настройка systemd-сервиса

1) Создать systemd-файл (например, /etc/systemd/system/classifier.service);

2) Ввести настройки запуска сервиса ***dotnet*** приложением с указанием параметров. Входной точкой приложения является ***Quarta.Classifier.WebApi.dll***;

3) Выполнить команду ***systemctl daemon-reload***, чтобы ввести сервис в действие.

4) Проверить состояние сервиса командой ***systemctl status classifier***;

Пример настроек ***classifier.service***

```
[Unit]
Description=Справочники

[Service]
WorkingDirectory=/usr/share/hosting/classifier
ExecStart=/usr/bin/dotnet Quarta.Classifier.WebApi.dll -s kip-env-config -migrations
Restart=always
RestartSec=10
SyslogIdentifier=dotnet-classifier
User=%ИМЯ СИСТЕМНОГО ПОЛЬЗОВАТЕЛЯ%
Environment=DOTNET_PRINT_TELEMETRY_MESSAGE=false
Environment=DOTNET_CLI_TELEMETRY_OPTOUT=false
Environment=ASPNETCORE_ENVIRONMENT=Production

[Install]
WantedBy=multi-user.target
```

2.4.4. Настройка nginx

1) Перейти в директорию ***/etc/nginx/conf.d*** (или аналогичную директорию для размещения динамический модулей ***nginx***);

2) Создать файл app-classifier-service.conf;

3) Внести настройки proxy_pass между внешним адресом сервера и внутренним;

4) Проверить корректность настроек командой ***nginx –t***;

5) Выполнить команду ***nginx reload*** если проверка прошла успешно.

Пример настроек ***app-classifier-service.conf***

```
server {
  listen 1300 ssl;

  location / {
    proxy_pass https://127.0.0.1:1300;
  }
}
```

Адрес proxy_pass должен соответствовать значению поля ApplicationUrl указанного в ***appSettings.json***.

2.4.5. Настройка SPA-приложения

1) Перейти в директорию дистрибутива quarta-classifier-web-app/config/clients/default;

2) Открыть файл configuration.json;

3) Внести настройки SPA-приложения.

Пример настроек ***configuration.json***

```
{
  "modules": {
    "classifier": {
      "serverUrl": "https://x.x.x.x", //внешний хост-адрес сервиса «Справочники»
      "baseUrl": "/app/classifier", //базовый url приложения
      "fileStorage": {
        "fileStorageUrl": "https://x.x.x.x/api/file-storage-proxy/",
        "maxAllowedFileSize": 1073741824
      }
    }
  },
  "authentication": {
```

```
            "authority": "https://x.x.x.x", //внешний хост адрес бэкенд-приложения сервиса Хаб
            "redirect_uri": "https://x.x.x.x/callback",  //внешний адрес переадресации
            "post_logout_redirect_uri": "https://x.x.x.x/login", //внешний адрес страницы логина внутри приложения
            "silent_redirect_uri": "https://x.x.x.x/silent-callback", //внешний адрес автообновления токена
            "client_id": "", //идентификатор клиента (Режим «Клиенты» в сервисе Хаб)
               "response_type": "id_token token",
               "scope": " openid  profile identityServer  classifier", //перечень доступных ресурсов.

               "useLocalStorage": false,
               "automaticSilentRenew": true
            }
        }
```

2.5. Установка и развертывание сервиса «Уведомления»

2.5.1. Подготовка БД

1) Подключиться по ***ssh*** к серверу, на котором развернута СУБД;
2) Выполнить команду ***psql***.
3) Выполнить команду create database app.notifications.

2.5.2. Настройка бэкенд-приложения

1) Скопировать дистрибутив сервиса в директорию сервера (например, ***/usr/share/hosting/notifications)***;
2) Создать пользователя, под которым будет запускаться приложение. Пользователь должен иметь полный доступ к директории с приложением и права открывать сокеты;
3) Перейти в директорию дистрибутива ***/clients/default***;
4) Выполнить настройку ***appsettings.json***.

```
{
 "ApplicationUrls": [
  "https://localhost:1400" //адрес который займет веб-сервер Kestrel
 ],
 "AuthServer": {
```

```
    "Url": "https://x.x.x.x", //хост приложение сервиса Хаб. Может
указываться внешний или внутренний адрес
    "ApiName": "", //идентификатор данного сервиса (режим «Ресурсы»
сервиса Хаб)
    "ApiSecret": "", //пароль данного сервиса (режим «Ресурсы» сервиса
Хаб, код авторизации)
    "ClaimType": "uri://schemas.quarta.su/permission-claim-type",
    "AccessEventEndpointName": "access_event_endpoint"
  },
  "ConnectionStrings": {
    "DefaultConnection": "Server=localhost; Database=app.notifications; User
ID=postgres; Password=postgres"
  },
  "EventBus": {
    "Enabled": true,
    "Broker": "app.workspace",
    "RetryCount": 10,
    "QueueName": "app.notifications",
    "CommandQueueName":"app.notifications",
    "ExchangeType": "direct",
    "BusAccess": {
      "Host": "x.x.x.x", //ИП адрес RabbitMQ
      "UserName": "user", //логин
      "Password": "password", //пароль
      "RetryCount": 10
    }
  },
  "CorsOrigins": [
    "^http[s]?:\\/\\/x\\.x\\.x\\.x(:\\d{1,6})?$",
    "^http[s]?:\\/\\/localhost(:\\d{1,6})?$"
  ],
  "Logging": {
    "LogLevel": {
      "Default": "Warning",
      "System": "Warning",
      "Microsoft": "Warning"
    }
  },
  "Smtp": {
    "Host": "", //имя хоста почтового сервера
    "Port": 465, //порт
```

```
    "IsSsl": false, //использоваться ли ssl при подключении
    "Email": "", //адрес, с которого отправлять сообщение
    "UserName": "", //логин подключения к почте
    "Password": "", /пароль подключения к почте
    "CheckCertificateRevocation": false //проверять ли ssl сертификат на
актуальность
  },
  "RabbitUserId": "E34D0E27-4176-469F-A5D7-BC34BC922510"
}
```

2.5.3. Настройка systemd-сервиса

1) Создать systemd-файл (например, /etc/systemd/system/ classifier.service);

2) Ввести настройки запуска сервиса ***dotnet*** приложением с указанием параметров. Входной точкой приложения является ***Quarta. Classifier.WebApi.dll***;

3) Выполнить команду ***systemctl daemon-reload***, чтобы ввести сервис в действие;

4) Проверить состояние сервиса командой ***systemctl status classifier***.

Пример настроек ***notification.service***

```
[Unit]
Description=Уведомления

[Service]
WorkingDirectory=/usr/share/hosting/notifications
ExecStart=/usr/bin/dotnet Quarta.Notification.WebApi.dll -skip-env-config -
migrations
Restart=always
RestartSec=10
SyslogIdentifier=dotnet-notification
User=%ИМЯ СИСТЕМНОГО ПОЛЬЗОВАТЕЛЯ%
Environment=DOTNET_PRINT_TELEMETRY_MESSAGE=false
Environment=DOTNET_CLI_TELEMETRY_OPTOUT=false
Environment=ASPNETCORE_ENVIRONMENT=Production

[Install]
WantedBy=multi-user.target
```

2.5.4. Настройка nginx

1) Перейти в директорию ***/etc/nginx/conf.d*** (или аналогичную директорию для размещения динамический модулей ***nginx***);

2) Создать файл ***app-notification-service.conf***;

3) Внести настройки proxy_pass между внешним адресом сервера и внутренним;

4) Проверить корректность настроек командой ***nginx –t***;

5) Выполнить команду ***nginx reload***, если проверка прошла успешно.

Пример настроек ***app-notification-service.conf***

```
server {
   listen 1400 ssl;

   location / {
      proxy_pass https://127.0.0.1:1400;
   }
}
```

Адрес proxy_pass должен соответствовать значению поля ApplicationUrl указанного в ***appSettings.json***.

2.5.5. Настройка SPA-приложения

1) Перейти в директорию дистрибутива quarta-notification-web-app/config/clients/default;

2) Открыть файл configuration.json;

3) Внести настройки SPA-приложения.

Пример настроек ***configuration.json***

```
{
   "modules": {
   "nt": {
     "serverUrl": "https://x.x.x.x", //внешний адрес бэкенд-приложения «Уведомления»
      "baseUrl": "/app/notification" //базовый url приложения
   }
   },
   "authentication": {
```

```
        "authority": "https://x.x.x.x", //внешний хост адрес бэкенд-приложения сервиса Хаб
        "redirect_uri": "https://x.x.x.x/callback",  //внешний адрес переадресации
        "post_logout_redirect_uri": "https://x.x.x.x/login", //внешний адрес страницы логина внутри приложения
        "silent_redirect_uri": "https://x.x.x.x/silent-callback", //внешний адрес автообновления токена
        "client_id": "", //идентификатор клиента (Режим «Клиенты» в сервисе Хаб)
          "response_type": "id_token token",
          "scope": " openid  profile identityServer  notification", //перечень доступных ресурсов.
          "useLocalStorage": false,
          "automaticSilentRenew": true
        }
    }
```

2.6. Установка и развертывание сервиса «Хранение и оборот документов»

2.6.1. Подготовка БД

1) Подключиться по ***ssh*** к серверу, на котором развернута СУБД;
2) Выполнить команду ***psql***;
3) Выполнить команду create database app.ds;

2.6.2. Настройка бэкенд-приложения

1) Скопировать дистрибутив сервиса в директорию сервера (например, ***/usr/share/hosting/ds***);
2) Создать пользователя, под которым будет запускаться приложение. Пользователь должен иметь полный доступ к директории с приложением и права открывать сокеты;
3) Перейти в директорию дистрибутива ***/clients/default***;
4) Выполнить настройку ***appsettings.json***.

```
{
 "ApplicationUrls": [ "https://localhost:1500" ],

 "ConnectionStrings": {
  "DefaultConnection": "Server=localhost; Database=app.ds; User ID=postgres; Password=postgres"
```

```
},
"AuthServer": {
  "Url": "https://x.x.x.x", //хост приложение сервиса Хаб. Может
указываться внешний или внутренний адрес
  "ApiName": "", //идентификатор данного сервиса (режим «Ресурсы»
сервиса Хаб)
  "ApiSecret": "", //пароль данного сервиса (режим «Ресурсы» сервиса
Хаб, код авторизации)
  "ClaimType": "uri://schemas.quarta.su/permission-claim-type",
  "AccessEventEndpointName": "access_event_endpoint"
},
"EventBus": {
  "Enabled": true,
  "Broker": "app.workspace",
  "RetryCount": 10,
  "QueueName": "app.ds",
  "CommandQueueName":"app.ds",
  "ExchangeType": "direct",
  "BusAccess": {
    "Host": "x.x.x.x", //ip-адрес RabbitMQ
    "UserName": "user", //логин
    "Password": "password", //пароль
    "RetryCount": 10
  }
},
"Logging": {
  "LogLevel": {
    "Default": "Warning",
    "System": "Warning",
    "Microsoft": "Warning"
  }
},
"OnlyofficeUrl": "https://x.x.x.x", //адрес веб-сервера OnlyOffice
"FileStorage": {
  "Url": http://x.x.x.x//адрес файлового хранилища.
},
"CorsOrigins": [
  "^http[s]?:\\/\\/x\\.x\\.x\\.x(:\\d{1,7})?$",
  "^http[s]?:\\/\\/localhost(:\\d{1,7})?$"
],
"ApacheTikaHost": "http://x.x.x.x", //адрес апатч тика
```

```
  "FullTextSearchUpdate": { //настройки полнотекстового поиска
    "IsWorkrOn": true,
    "RecordsToProcess": 50,
    "ThreadCount": 10
  }
}
```

2.6.3. Настройка systemd-сервиса

1) Создать systemd-файл (например, ***/etc/systemd/system/ds.service***);

2) Ввести настройки запуска сервиса ***dotnet*** приложением с указанием параметров. Входной точкой приложения является ***Quarta.DS.WebApi.dll***;

3) Выполнить команду ***systemctl daemon-reload***, чтобы ввести сервис в действие.

4) Проверить состояние сервиса командой ***systemctl status ds***.

Пример настроек ***ds.service***

```
[Unit]
Description=Файловое хранилище

[Service]
WorkingDirectory=/usr/share/hosting/ds
ExecStart=/usr/bin/dotnet Quarta.DS.WebApi.dll -skip-env-config -migrations
Restart=always
RestartSec=10
SyslogIdentifier=dotnet-ds
User=%ИМЯ СИСТЕМНОГО ПОЛЬЗОВАТЕЛЯ%
Environment=DOTNET_PRINT_TELEMETRY_MESSAGE=false
Environment=DOTNET_CLI_TELEMETRY_OPTOUT=false
Environment=ASPNETCORE_ENVIRONMENT=Production

[Install]
WantedBy=multi-user.target
```

2.6.4. Настройка nginx

1) Перейти в директорию ***/etc/nginx/conf.d*** (или аналогичную директорию для размещения динамический модулей ***nginx***);

2) Создать файл app-ds-service.conf;

3) Внести настройки proxy_pass между внешним адресом сервера и внутренним;

4) Проверить корректность настроек командой ***nginx –t***;

5) Выполнить команду ***nginx reload***, если проверка прошла успешно;

Пример настроек app-ds-service.conf

```
server {
   listen 1500 ssl;

   location / {
      proxy_pass https://127.0.0.1:1500;
   }
}
```

Адрес proxy_pass должен соответствовать значению поля ApplicationUrl указанного в ***appSettings.json***.

2.6.5. Настройка SPA-приложения

1) Перейти в директорию дистрибутива quarta-ds-web-app/config/clients/default;

2) Открыть файл configuration.json;

3) Внести настройки SPA-приложения.

Пример настроек ***configuration.json***

```
{
   "modules": {
      "ds": {
         "serverUrl": "https://x.x.x.x", //внешний адрес бэкенд-приложения «Оборот и хранение документов»
         "baseUrl": "/app/ds", //базовый url приложения
         "fileStorage": {
            "fileStorageUrl": "https://x.x.x.x/api/file-storage-proxy/",
            "maxAllowedFileSize": 1073741824
         },
         "options": {          //настройки приложения
            "fileLibrary": {
               "tabs": ["allFiles"],
               "cardTabs": ["main", "approvals", "history"],
```

```
                    "buttons": {
                        "addFile": false,
                        "addFolder": true
                    }
                }
            }
        },
        "classifier": {
            "serverUrl": https://x.x.x.x//внешний адрес приложения
«Справочники»
        },
        "onlyoffice": {
            "serverUrl": https://x.x.x.x//внешний адрес сервиса Only Office.
        }
    },
    "authentication": {
    "authority": "https://x.x.x.x", //внешний хост адрес бэкенд-приложения
сервиса Хаб
    "redirect_uri": "https://x.x.x.x/callback",  //внешний адрес переадресации
    "post_logout_redirect_uri": "https://x.x.x.x/login", //внешний адрес
страницы логина внутри приложения
    "silent_redirect_uri": "https://x.x.x.x/silent-callback", //внешний адрес
автообновления токена
    "client_id": "", //идентификатор клиента (Режим «Клиенты» в сервисе
Хаб)
        "response_type": "id_token token",
        "scope": " openid  profile identityServer  ds classifier", //перечень
доступных ресурсов.
        "useLocalStorage": false,
        "automaticSilentRenew": true
    }
}
```

2.7. Установка и развертывание сервиса «ОШС»

2.7.1. Подготовка БД

1) Подключиться по ***ssh*** к серверу, на котором развернута СУБД;
2) Выполнить команду ***psql***;
3) Выполнить команду create database app.ss.

2.7.2. Настройка бэкенд-приложения

1) Скопировать дистрибутив сервиса в директорию сервера (например, ***/usr/share/hosting/ss***);

2) Создать пользователя, под которым будет запускаться приложение. Пользователь должен иметь полный доступ к директории с приложением и права открывать сокеты;

3) Перейти в директорию дистрибутива ***/clients/default***;

4) Выполнить настройку ***appsettings.json***.

```
{
 "ApplicationUrls": [
  http://localhost:1200 //адрес который займет веб-сервер Kestrel
 ],
 "AuthServer": {
  "Url": "https://x.x.x.x", //хост приложение Хаб. Может указываться внешний или внутренний адрес
  "ApiName": "", //идентификатор данного сервиса (режим «Ресурсы» сервиса Хаб)
  "ApiSecret": "", //пароль данного сервиса (режим «Ресурсы» сервиса Хаб, код авторизации)
  "ClaimType": "uri://schemas.quarta.su/permission-claim-type",
  "AccessEventEndpointName": "access_event_endpoint"
 },
 "ConnectionStrings": {
  "DefaultConnection": "Server=localhost; Database=app.ss; User ID=postgres; Password=postgres"
 },
 "EventBus": {
  "Enabled": true,
  "Broker": "app.workspace",
  "RetryCount": 10,
  "QueueName": "app.ss",
  "CommandQueueName":"app.ss",
  "ExchangeType": "direct",
  "BusAccess": {
   "Host": "x.x.x.x", //ip-адрес RabbitMQ
   "UserName": "user", //логин
   "Password": "password", //пароль
   "RetryCount": 10
  }
```

```
  },
  "CorsOrigins": [
    "^http[s]?:\\/\\/x\\.x\\.x\\.x(:\\d{1,7})?$",
    "^http[s]?:\\/\\/localhost(:\\d{1,7})?$"
  ],
  "Logging": {
    "LogLevel": {
      "Default": "Warning",
      "System": "Warning",
      "Microsoft": "Warning"
    }
  },
  "FileStorage": {
    "Url": http://x.x.x.x//адрес файлового хранилища.
  }
}
```

2.7.3. Настройка systemd-сервиса

1) Создать systemd-файл (например, ***/etc/systemd/system/ss.service***);

2) Ввести настройки запуска сервиса ***dotnet*** приложением с указанием параметров. Входной точкой приложения является ***Quarta.SS.WebApi.dll***;

3) Выполнить команду ***systemctl daemon-reload***, чтобы ввести сервис в действие.

4) Проверить состояние сервиса командой ***systemctl status ss***.

Пример настроек ***ss.service***

```
[Unit]
Description=ОШС

[Service]
WorkingDirectory=/usr/share/hosting/ss
ExecStart=/usr/bin/dotnet Quarta.SS.WebApi.dll -skip-env-config -migrations
Restart=always
RestartSec=10
SyslogIdentifier=dotnet-ss
User=%ИМЯ СИСТЕМНОГО ПОЛЬЗОВАТЕЛЯ%
Environment=DOTNET_PRINT_TELEMETRY_MESSAGE=false
Environment=DOTNET_CLI_TELEMETRY_OPTOUT=false
```

```
Environment=ASPNETCORE_ENVIRONMENT=Production

[Install]
WantedBy=multi-user.target
```

2.7.4. Настройка nginx

1) Перейти в директорию ***/etc/nginx/conf.d*** (или аналогичную директорию для размещения динамический модулей ***nginx***);
2) Создать файл app-ss-service.conf;
3) Внести настройки proxy_pass между внешним адресом сервера и внутренним;
4) Проверить корректность настроек командой ***nginx –t***;
5) Выполнить команду ***nginx reload***, если проверка прошла успешно.

Пример настроек ***app-ss-service.conf***

```
server {
   listen 1200 ssl;

   location / {
      proxy_pass https://127.0.0.1:1200;
   }
}
```

Адрес proxy_pass должен соответствовать значению поля ApplicationUrl указанного в ***appSettings.json***.

2.8. Установка и развертывание сервиса «Оболочка»

2.8.1. Развертывание дистрибутива

1) Скопировать дистрибутив сервиса в директорию сервера (например, ***/usr/share/hosting/ws***);
2) Создать пользователя, под которым будет запускаться приложение. Пользователь должен иметь полный доступ к директории с приложением;
3) Перейти в директорию дистрибутива ***/clients/default***.

2.8.2. Настройка nginx

1) Перейти в директорию ***/etc/nginx/conf.d*** (или аналогичную директорию для размещения динамический модулей ***nginx***);

2) Создать файл app-ws-service.conf;

3) Внести настройки proxy_pass между внешним адресом сервера и внутренним;

4) Проверить корректность настроек командой ***nginx*** –t;

5) Выполнить команду ***nginx reload***, если проверка прошла успешно.

Пример настроек ***app-ws-service.conf***

(указанные хосты для proxy_pass ссылаются на внутренние адреса сервисов (может также использоваться публичный адрес)

```
listen 80 ssl;
    error_page  497 https://$host:80$request_uri;
    root /usr/share/hosting/ws;
    location / {
        try_files $uri $uri/ /index.html;
    }
  #”Сервис Хранилище Документов»
  location /assets/ds {
        proxy_pass https://localhost:1500;
    }
# «Сервис ОШС»
  location /assets/ss {
        proxy_pass https://localhost:1200;
    }
# «Сервис Уведомления»
location /assets/nt {
        proxy_pass https://localhost:1400;
    }
```

2.8.3. Настройка SPA-приложения

1) Перейти в директорию дистрибутива ***quarta-ws-web-app/config/***;

2) Открыть файл configuration.json;

3) Внести настройки SPA-приложения.

Пример настроек ***configuration.json***

```
{
  "modules": {
    "ds": {
      "serverUrl": " https://x.x.x.x", //Публичный хост «Сервис Хранилище документов»
      "path": "ds",
      "data": {
        "classifiers": [
          {
            "label": "Виды документов",
            "route": "classifier/document-sub-types",
            "readClaim": "ds/classifier/document_sub_types/read"
          },
          {
            "label": "Комиссии",
            "route": "classifier/comissions",
            "readClaim": "ds/classifier/commissions/read"
          },
          {
            "label": "Роли согласующих лиц",
            "route": "classifier/participant-roles",
            "readClaim": "ds/classifier/participant_roles/read"
          },
          {
            "label": "Шаблоны маршрутов согласования",
            "route": "document-flow"
          }
        ]
      },
      "remoteEntry": "https://x.x.x.x/remoteEntry.js", //Публичный хост «Сервис Хранилище документов»
      "exposedModule": "./FeaturesModule",
      "key": "FeaturesModule",
      "name": "Documentation",
      "fileStorage": {
        "fileStorageUrl": "https://x.x.x.x./api/file-storage-proxy
        "maxAllowedFileSize": 1073741824
      }
    },
    "ss": {
      "serverUrl": "https://x.x.x.x.", //Публичный хост «Сервис ОШС»
```

```
                "path": "ss",
                "data": {
                  "classifiers": [
                    {
                      "label": "Должности",
                      "route": "classifier/positions",
                      "readClaim": "ss/classifier/position/read"
                    },
                    {
                      "label": "Подразделения",
                      "route": "classifier/structural-units",
                      "readClaim": "ss/classifier/structural_unit/read"
                    },
                    {
                      "label": "Сведения об отсутствии на рабочем месте",
                      "route": "classifier/temporarily-vacants",
                      "readClaim": "ss/classifier/temporarily_vacant/read"
                    },
                    {
                      "label": "Сотрудники",
                      "route": "classifier/employees",
                      "readClaim": "ss/classifier/employee/read"
                    }
                  ]
                },
                "remoteEntry": "https://x.x.x.x./remoteEntry.js” , //Публичный хост
«Сервис ОШС»
                "exposedModule": "./FeaturesModule",
                "key": "FeaturesModule",
                "name": "StaffStructure"
              },
              "nt": {
                "serverUrl": "https://x.x.x.x.", //Публичный хост «Сервис
Уведомлений»
                "path": "notifications",
                "remoteEntry": "http://x.x.x.x./remoteEntry.js", //Публичный хост
«Сервис Уведомлений»
                "exposedModule": "./FeaturesModule",
                "key": "FeaturesModule",
                "name": "Notifications"
              },
```

```
                "classifier": {
                    "serverUrl": “https://x.x.x.x” //Публичный хост «Сервис
Справочников»
                },
                "onlyoffice": {
                    "serverUrl": "https://x.x.x.x"  //Публичный хост «Only Office»

                }
            },
            "authentication": {
            "authority": "https://x.x.x.x", //внешний хост адрес бэкенд-приложения
“Хаб»
            "redirect_uri": "https://x.x.x.x/callback",  //внешний адрес переадресации
приложения
            "post_logout_redirect_uri": "https://x.x.x.x/login", //внешний адрес
страницы логина внутри приложения
            "silent_redirect_uri": "https://x.x.x.x/silent-callback", //внешний адрес
автообновления токена
            "client_id": "", //идентификатор клиента (Режим «Клиенты» в Хабе)
                "response_type": "id_token token",
                "scope": "openid identityServer ds ss bi notification classifier
integration", //перечень доступных ресурсов (указывается в «Хабе», режим
«Ресурсы»
                "useLocalStorage": false,
                "automaticSilentRenew": true
            }
        }
```

3. Обновление программного обеспечения

3.1. Обновление сервиса «Администрирование и контроль доступа (Хаб)»

1) Распаковать дистрибутив с обновлением во временную папку.

2) Перенести существующие настройки (значения) из старого конфигурационного файла в новый.

- Конфигурационный файл бэкэнда сервиса находится в директории ./Clients/Default/appsettings.json
- Конфигурационный файл фронтэнда сервиса находится в директории ./wwwroot/app/assets/configurations/clients/default/configuration.json

3) Удалить директорию с сервисом, на место удаленной директории перенести директорию с настроенным обновлением.

4) Выдать разрешение на папку для пользователя, под которым настроена служба. Пользователь должен быть владельцем папки с правами на чтение / запись / выполнение.

5) Перезапустить службу.

3.2. Обновление сервиса «Файловое хранилище»

1) Распаковать дистрибутив с обновлением во временную папку.

2) Перенести существующие настройки (значения) из старого конфигурационного файла в новый.

- Конфигурационный файл бэкэнда сервиса находится в директории ./appsettings.json

3) Удалить директорию с сервисом, на место удаленной директории перенести директорию с настроенным обновлением.

4) Выдать разрешение на папку для пользователя, под которым настроена служба. Пользователь должен быть владельцем папки с правами на чтение / запись / выполнение.

5) Перезапустить службу.

3.3. Обновление сервиса «Справочники»

1) Распаковать дистрибутив с обновлением во временную папку.

2) Перенести существующие настройки (значения) из старого конфигурационного файла в новый.

- Конфигурационный файл бэкэнда сервиса находится в

директории ./Clients/Default/appsettings.json

- Конфигурационный файл фронтэнда сервиса находится в директории ./quarta-classifier-web-app/config/clients/default/configuration.json

3) Удалить директорию с сервисом, на место удаленной директории перенести директорию с настроенным обновлением.

4) Выдать разрешение на папку для пользователя, под которым настроена служба. Пользователь должен быть владельцем папки с правами на чтение / запись / выполнение.

5) Перезапустить службу.

3.4. Обновление сервиса «Уведомления»

1) Распаковать дистрибутив с обновлением во временную папку.

2) Перенести существующие настройки (значения) из старого конфигурационного файла в новый.

- Конфигурационный файл бэкэнда сервиса находится в директории ./Clients/Default/appsettings.json
- Конфигурационный файл фронтэнда сервиса находится в директории ./quarta-notification-web-app/config/clients/default/configuration.json

3) Удалить директорию с сервисом, на место удаленной директории перенести директорию с настроенным обновлением.

4) Выдать разрешение на папку для пользователя, под которым настроена служба. Пользователь должен быть владельцем папки с правами на чтение / запись / выполнение.

5) Перезапустить службу.

3.5. Обновление сервиса «Хранение и оборот документов»

1) Распаковать дистрибутив с обновлением во временную папку.

2) Перенести существующие настройки (значения) из старого конфигурационного файла в новый.

- Конфигурационный файл бэкэнда сервиса находится в директории ./Clients/Default/appsettings.json
- Конфигурационный файл фронтэнда сервиса находится в директории ./quarta-ds-web-app/config/clients/default/configuration.json

3) Удалить директорию с сервисом, на место удаленной директории перенести директорию с настроенным обновлением.

4) Выдать разрешение на папку для пользователя, под которым настроена служба. Пользователь должен быть владельцем папки с правами на чтение / запись / выполнение.

5) Перезапустить службу.

3.6. Обновление сервиса «ОШС»

1) Распаковать дистрибутив с обновлением во временную папку.

2) Перенести существующие настройки (значения) из старого конфигурационного файла в новый.

- Конфигурационный файл бэкэнда сервиса находится в директории ./Clients/Default/appsettings.json

3) Удалить директорию с сервисом, на место удаленной директории перенести директорию с настроенным обновлением.

4) Выдать разрешение на папку для пользователя, под которым настроена служба. Пользователь должен быть владельцем папки с правами на чтение / запись / выполнение.

5) Перезапустить службу.

3.7. Обновление сервиса «Оболочка»

1) Распаковать дистрибутив с обновлением во временную папку.

2) Перенести существующие настройки (значения) из старого конфигурационного файла в новый.

3) Конфигурационный файл фронтэнда сервиса находится в директории ./config/configuration.json

4) Удалить директорию с сервисом, на место удаленной директории перенести директорию с настроенным обновлением.

5) Выдать разрешение на папку для пользователя, под которым запущен публикующий этот сервис HTTP-сервер. Пользователь должен быть владельцем папки с правами на чтение / запись / выполнение.

4. Проверка, восстановление и поддержание работоспособности программного обеспечения

4.1. Методы проверки работоспособности рабочих станций

Чтобы проверить работоспособность рабочей станции (клиентской части Системы), требуется выполнить следующие действия:

– запустить рабочую станцию;
– запустить web-браузер;
– в адресной строке ввести адрес сервера Системы;
– в окне авторизации ввести логин и пароль (login: admin_user, пароль: admin), нажать на кнопку «Войти».

Система работоспособна, если в результате выполнения действий отображается главная страница.

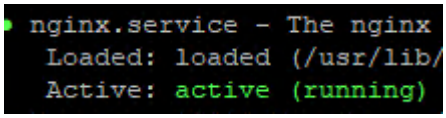
4.2. Методы проверки работоспособности сервера

Чтобы проверить работоспособность сервера, требуется выполнить следующие действия:

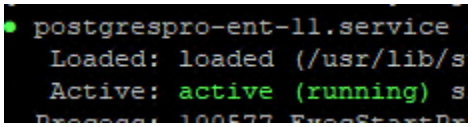
– убедиться в том, что службы Postgres Pro, nginx, MongoDB и сервисов Системы работают.

Для этого убеждаемся, что их статус active, командами:

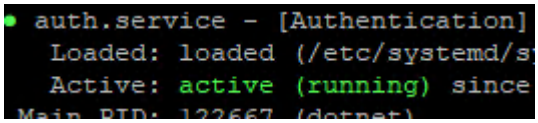
systemctl status *nginx*



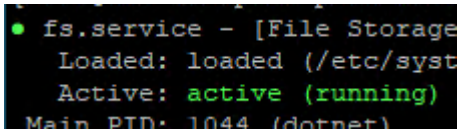
systemctl status postgrespro-ent-14



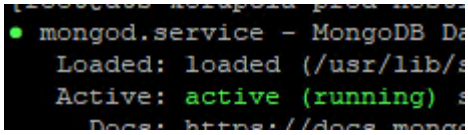
systemctl status *auth* (в зависимости от того, какое имя службе модуля хаба выдали);



systemctl status *fs* (в зависимости от того, какое имя службе модуля файлового хранилища выдали);



systemctl status *mongod* (в зависимости от того, какое имя службе модуля файлового хранилища выдали).



– убедиться в том, что Postgres Pro доступен. Для этого достаточно выполнить команду psql и в запустившемся интерактивном терминале Postgres Pro выполнить команду: \conninfo. При успешном соединении отобразится соответствующее сообщение, например:

```
postgres=# \conninfo
You are connected to database "postgres" as user "postgres" via socket in "/var/run/postgresql" at port "54
32".
postgres=#
```

– убедиться в том, что Postgres Pro, nginx, MongoDB и Система открыли все настроенные порты:

netstat -tnulp

```
[root@DTS-KORUPCIA-APP ~]# netstat -tnulp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:5000          0.0.0.0:*               LISTEN      51763/dotnet
tcp        0      0 127.0.0.1:5001          0.0.0.0:*               LISTEN      51763/dotnet
tcp        0      0 0.0.0.0:1040            0.0.0.0:*               LISTEN      15021/nginx: master
tcp        0      0 127.0.0.1:5040          0.0.0.0:*               LISTEN      1044/dotnet
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1147/sshd
tcp        0      0 0.0.0.0:5432            0.0.0.0:*               LISTEN      100581/postgres
tcp        0      0 0.0.0.0:1080            0.0.0.0:*               LISTEN      15021/nginx: master
tcp        0      0 127.0.0.1:5080          0.0.0.0:*               LISTEN      1043/dotnet
tcp        0      0 127.0.0.1:17017         0.0.0.0:*               LISTEN      1347/mongod
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1336/master
tcp        0      0 0.0.0.0:1050            0.0.0.0:*               LISTEN      15021/nginx: master
tcp        0      0 0.0.0.0:1090            0.0.0.0:*               LISTEN      15021/nginx: master
tcp        0      0 127.0.0.1:5090          0.0.0.0:*               LISTEN      1041/dotnet
```

4.3. Методы проверки работоспособности базы данных

В проверку работоспособности базы данных входит:

– проверка физической целостности базы данных;

– проверка сохранения введенных данных.

4.3.1. Проверка физической целостности базы данных

Проверка физической целостности базы данных проводится подсчётом контрольных сумм на файлах баз данных кластера.

– Для этого следует предварительно выключить службу СУБД:

systemctl stop postgrespro-ent-14

– Запустить проверку:

pg_verify_checksums -D /var/lib/pgpro/ent-14/data/

```
Проверка контрольных сумм завершена
Версия контрольных сумм данных: 1
Просканировано файлов: 6147
Просканировано блоков: 13774
Неверные контрольные суммы: 0
```

– Проверить целостность базы данных mongodb:

– Подключиться к mongo и выполнить:

use filestorage

где filestorage наше имя базы данных.

– Открыть список коллекций:

db.getCollectionInfos();

– Далее для каждой коллекции выполнить проверку

db.files.validate({full:true})

В результате valid должен иметь значение true

```
{
        "ns" : "filestorage.files",
        "nInvalidDocuments" : 0,
        "nrecords" : 90,
        "nIndexes" : 1,
        "keysPerIndex" : {
                "_id_" : 90
        },
        "indexDetails" : {
                "_id_" : {
                        "valid" : true
                }
        },
        "valid" : true,
        "repaired" : false,
        "warnings" : [ ],
        "errors" : [ ],
        "extraIndexEntries" : [ ],
        "missingIndexEntries" : [ ],
        "corruptRecords" : [ ],
        "ok" : 1
}
```

4.3.2. Проверка сохранения введенных данных

Для проверки сохранения данных, вносимых в базу данных Системы через браузер, нужно выполнить следующие действия:

1) запустить рабочую станцию;
2) открыть браузер;
3) в адресной строке ввести адрес сервера Системы;
4) в окне авторизации ввести логин и пароль, нажать на кнопку «Войти»;

5) на главной странице выбрать раздел **Библиотека документов**;
6) добавить документ;
7) закрыть браузер;
8) повторно выполнить пп. 2-5;
9) в списке найти добавленный документ;
10) удалить ранее добавленный документ.

Сохранение файла документа свидетельствует о работоспособности БД Системы.

4.4. Методы восстановления работоспособности сервера

Работоспособность сервера, в случае его отказа, восстанавливается специалистами из подразделения технической поддержки. Если технических неисправностей в оборудовании сервера не обнаружено и операционная система сервера работает без сбоев, то для восстановления его работоспособности рекомендуется переустановить серверную часть Системы.

4.5. Методы восстановления работоспособности базы данных

4.5.1. Методы восстановления работоспособности базы данных под управлением СУБД Postgres Pro 14

Для обеспечения возможности восстановления базы данных (например, поврежденной) администратор должен периодически выполнять её резервное копирование.

4.5.1.1. Резервное копирование базы данных

1) Подключиться под пользователем **postgres**:

su postgres

2) Создать резервную копию базы данных «database» (здесь подставить имя исходной базы данных) в файл по пути /tmp/database.bak:

pg_dump database > /tmp/database.bak

В результате процедуры по указанному пути будет создан файл резервной копии.

Рекомендация. Для гарантирования сохранности данных файл рекомендуется скопировать (средствами операционной системы) на внешнее устройство (CD-диск, флэш-карту или др.).

4.5.1.2. Восстановление базы данных

1) Подключиться под пользователем postgres:

su postgres

2) Восстановить базу «database» из файла бэкапа /tmp/database.bak:

psql database < /tmp/database.bak

4.5.2. Методы восстановления работоспособности базы данных под управлением СУБД MongoDB

4.5.2.1. Резервное копирование базы данных

Для резервного копирования на сервере необходимо запустить процедуру mongodump. В качестве параметров следует указать порт (port), на котором запущена служба, базу данных (db) и путь к директории (out):

mongodump --port=27017 --db filestorage --out=/tmp/mongo

В результате процедуры по указанному пути будет создана директория с резервной копией базы данных.

4.5.2.2. Восстановление базы данных

Для резервного копирования на сервере необходимо запустить процедуру mongorestore. В качестве параметров указываем порт (port) и директорию с бэкапом (dir).

mongorestore --port=27017 --dir=/tmp/mongo/

4.6. Методы поддержания целостности базы данных

Целостность базы данных обеспечивается встроенными средствами СУБД Postgres Pro 14 и СУБД MongoDB 6.

Кроме того, целостность баз данных Системы обеспечивается на этапе ввода данных посредством заданных разработчиком правил.

4.7. Методы поддержания безопасности базы данных

Безопасность базы данных обеспечивается встроенными средствами СУБД Postgres Pro 14, СУБД MongoDB 6 и операционными системами серверов.

4.8. Методы диагностирования проблем обновления баз данных

Обновление баз данных осуществляются через утилиту Quarta.Migrator.exe

В процессе прогона миграционных скриптов и обновления функций, триггеров и др. в случае возникновения ошибок утилита логирует проблемы.

4.8.1. Диагностирование проблем в работе сервисов

В процессе работы сервисы записывают ошибки в лог.

Для диагностирования проблем необходим SSH Клиент (например, Putty или аналог).

1) Подключиться к серверу, на котором осуществляется хостинг проблемного сервиса.

Для входа требуется:

- Хост (ip сервера);
- Порт;
- Логин;
- Пароль.

2) Получить идентификатор интересующего сервиса.

Для этого можно воспользоваться командой получения перечня докер-образов:

docker ps -a

Пример:

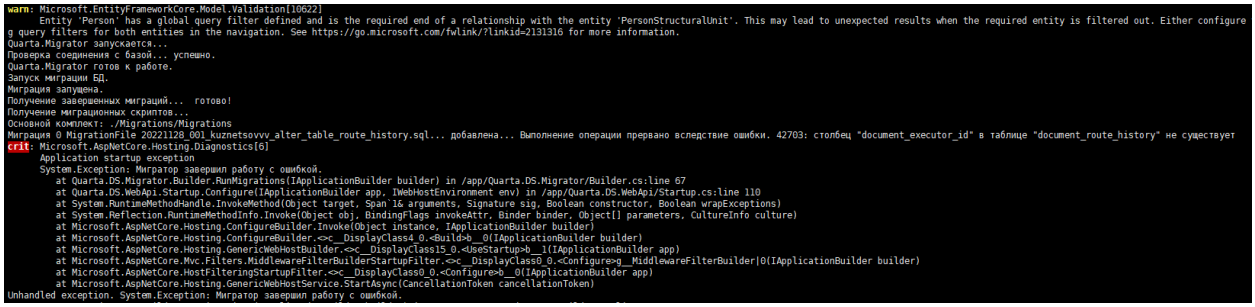
```
CONTAINER ID   IMAGE
5fa324f5c2f7   harbor.quarta.su/app-documentstorage/frontend:latest
a5998baf3ecd   harbor.quarta.su/app-documentstorage/app:latest
40ad2cab9706   harbor.quarta.su/app-integration-gosduma/app:latest
20305c5b0349   harbor.quarta.su/app-integration-gosduma/frontend:latest
ea3db8e85d49   harbor.quarta.su/app-notification/frontend:latest
32f44cc86e5d   harbor.quarta.su/app-notification/app:latest
da9c4c91f3f2   harbor.quarta.su/app-staffstructure/app:latest
4f5c1b57cc5f   harbor.quarta.su/app-staffstructure/frontend:latest
```

Например, необходимы логи сервиса «Хранение и оборот документов». Его имя: App-documentstorage, значит, идентификатор: a5998baf3ecd.

Если docker не используется, то логи доступны через команды вида journalctl -u %наименование сервиса% -n 100 -f

3) Получить логи сервиса с помощью команды команда:

docker logs a5998baf3ecd



В логах можно отследить предупреждения и ошибки работы .Net-сервисов в зависимости от настройки секции Logging конфигурационного файла appsettings.json.

Настройка фиксации только ошибок:

"Logging": {
  "LogLevel": {
    "Default": "Error",
    "System": " Error" } },

Настройка логирования предупреждений:

"Logging": {
  "LogLevel": {
    "Default": "Warning",
    "System": "Warning" } },

Настройка логирования максимальной информации (не рекомендуется, т.к. лог будет перегружен):

"Logging": {
  "LogLevel": {
    "Default": "Info",
    "System": " Info " } },

5. Администрирование программного обеспечения

Администрирование Системы обеспечивает:

- настройку и управление параметрами системы аутентификации пользователей, обеспечивающей доступ пользователей к ресурсам информационного портала;
- ведение списка пользователей, обеспечение регистрации пользователей информационного портала с предоставлением им ролей и прав доступа;
- ведение списка приложений и web-ресурсов;
- просмотр списка параметров работы Системы, настройка значений параметров для организаций группы компаний;
- ведение журналов событий аутентификации пользователей и проверок их прав доступа к информационным ресурсам, с возможностью формирования отчета об активности пользователей за период.

Задачи администрирования Системы выполняются в сервисе «Администрирование и контроль доступа (Хаб)».

Сервис Хаб предназначен для управления набором подключенных сервисов, управления правами доступа пользователей к Системе, авторизации и аутентификации пользователей, журналирования событий безопасности, а также настройки некоторых параметров функционирования Системы.

Сервис включает следующие режимы:

- Пользователи;
- Роли;
- Сотрудники;
- Константы;
- Клиенты;
- Ресурсы;
- События аутентификации;
- События безопасности;
- Активность пользователей.

5.1. Доступ к сервису

Для получения доступа к сервису Хаб:

1) в **адресной строке** браузера ввести адрес сервера. Адрес сервера имеет следующий вид – *http://адрес_веб-сервера* (пример: http://localhost:8080);

2) в окне **Авторизации** ввести логин и пароль пользователя с правами администратора *(по умолчанию доступен пользователь «test» с паролем «1»)*, нажать на кнопку **«Войти»**:

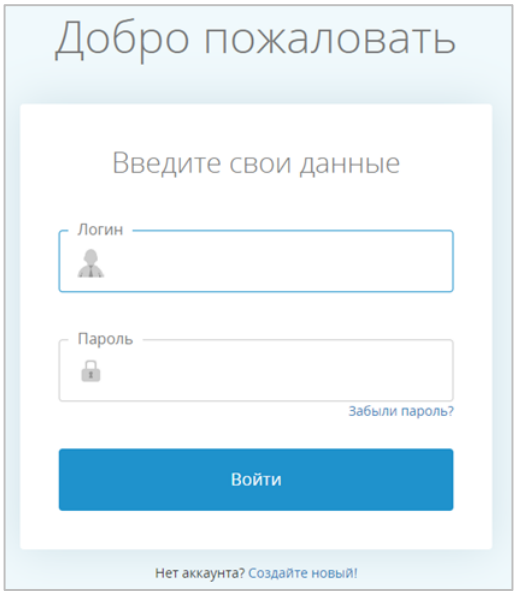


Рис. 5.1 – Окно авторизации

Для перехода в раздел «Администрирование» следует нажать кнопку профиля и в открывшемся меню выбрать **Администрирование**.

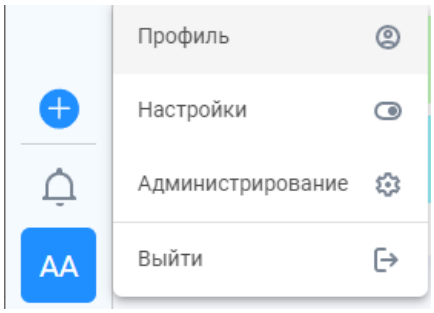


Рис. 5.2

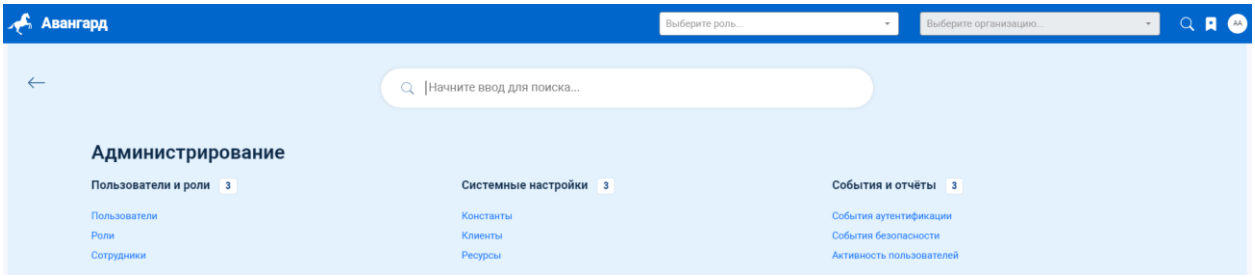


Рис. 5.3 – Главная страница сервиса администрирования

5.2. Управление доступом

К задачам управления доступом относятся:

– Настройка фронтенд-приложений и бэкенд-сервисов системы (клиентов, ресурсов);
– Настройка прав доступа к режимам Системы (формирование ролей, списка пользователей и сотрудников).

При запуске Системы выполняется сначала верификация клиента, потом авторизация пользователя. В ходе работы пользователя клиент (приложение) взаимодействует с ресурсами и сервисами.

5.2.1. Клиенты

Выбрать подраздел ***«Системные настройки» → Клиенты***. Откроется списочная форма Клиент-Приложений.

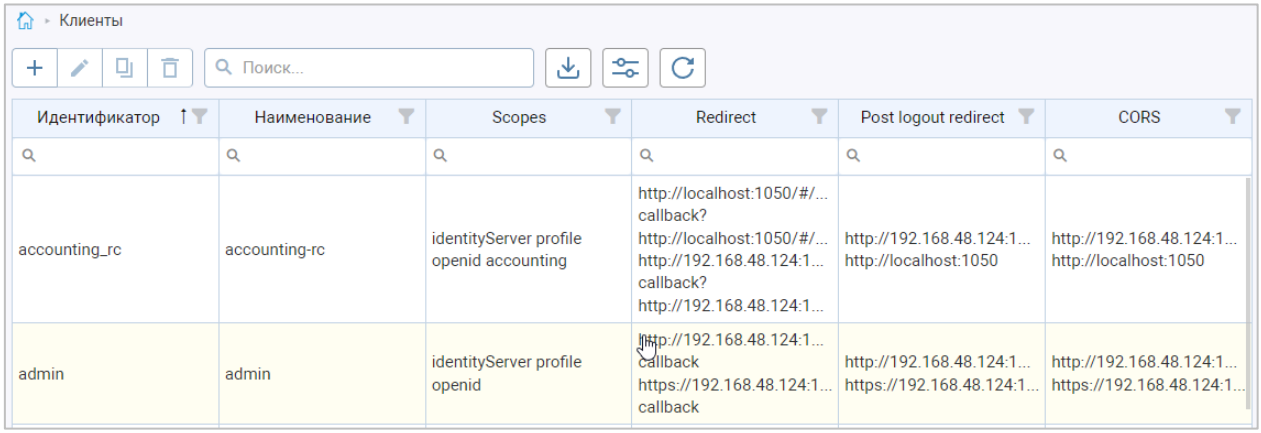


Рис. 5.4 – Клиенты

Списочная форма Клиент-Приложений содержит следующий набор сведений о настройках приложений:

1) **Идентификатор** – уникальный строковый идентификатор приложения;

2) **Наименование** – Читабельное представление (описание) приложения;

3) **Scopes** – Перечень разрешений, которые может запрашивать приложение;

4) **Redirect** – Перечень разрешенных переадресаций пользователя после прохода аутентификации;

5) **Post Logout redirect** – Перечень разрешенных переадресаций пользователя после того, как пользователь выходит из системы;

6) **CORS** – Перечень хостов, с которых приложению разрешено осуществлять запросы на аутентификацию/авторизацию пользователя.

Создание нового клиента:

Нажать «+» над списочной формой клиентов. Будет открыта форма заполнения данных:

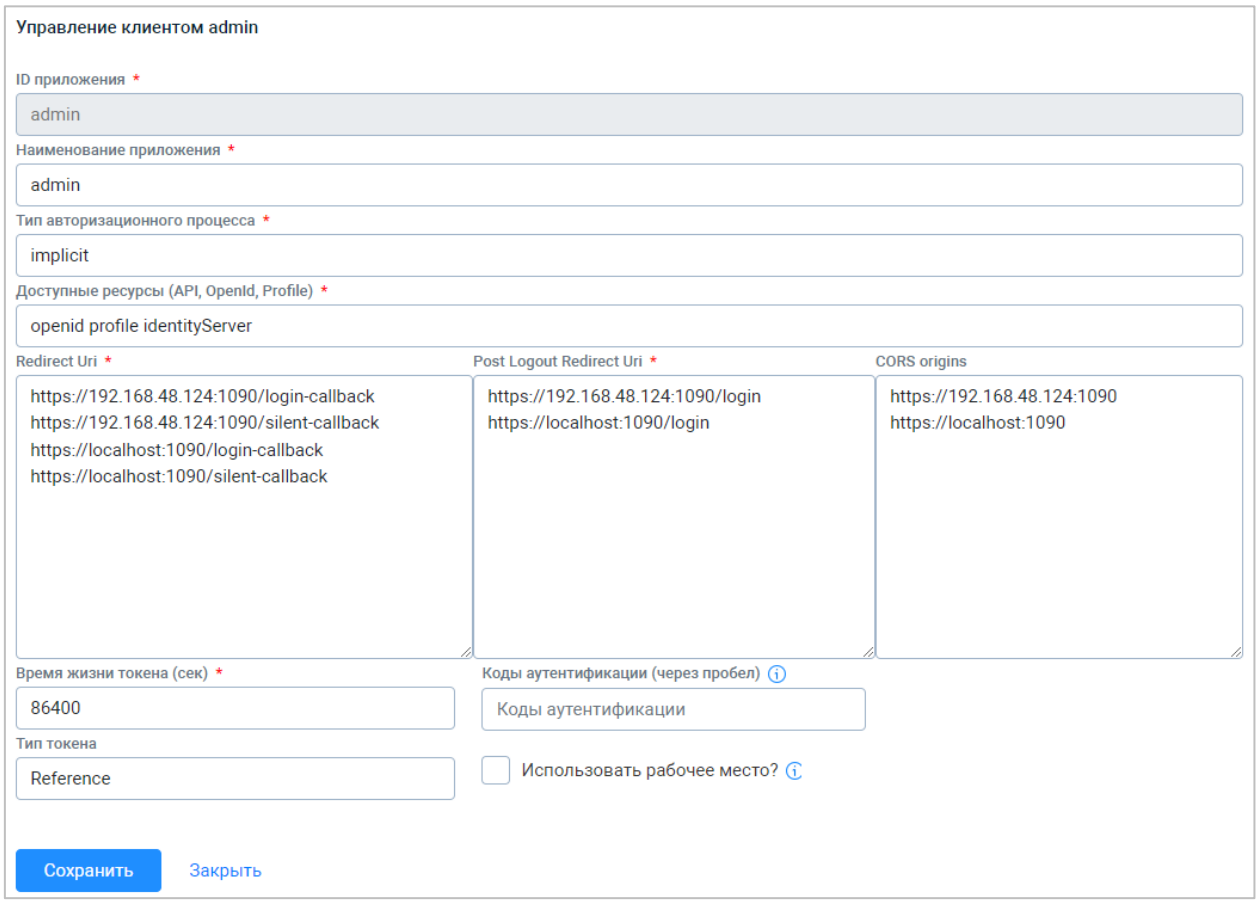


Рис. 5.5 – Описание клиента

Поля повторяют ранее описанные поля для списочной формы. Для всех полей, где возможен ввод нескольких значений, их порядок не имеет значения. Из особенностей отдельных полей:

Доступные ресурсы – каждый клиент должен содержать обязательные параметры: ***openid***, ***profile***. Остальные значения должны браться из списка идентификаторов Ресурсов, заполненных в режиме **«Ресурсы»**. Разделителем всех значений является символ «пробел».

Время жизни токена – заполняется исходя из требований к безопасности. Чем меньше интервал жизни токена, тем чаще осуществляется его автоматическое обновление, однако не рекомендуется делать его короче среднего времени сессии пользователя.

Тип токена – влияет на то, как будет передаваться токен авторизации из SPA-Приложения в Ресурс:

1) Reference: передаваться будет только обезличенный хэш-ключ, который требует подтверждения от сервиса Хаб;
2) JWT – самодостаточный токен.

Коды аутентификации – ключи для автоматизации отдельных аспектов процесса разработки, не предусмотрены для использования в реальных сценариях.

Требования к полям **Redirect Uri, Post Logout Redirect Uri** – разрешения адреса указываются в формате URL без указания возможных параметров или якорей: ***<схема>: [//<хост>[:<порт>]][/URL-путь>]***.

Пример корректно написанных URL:

https://auth.gd-workspace.ru/callback
http://192.168.48.124:1090/login
https://gd-workspace.ru

Требования к полю CORS: разрешенный адрес должен содержать только элементы: ***<схема>: [//<хост>[:<порт>]] (т.е. без путей, параметров и якорей)***.

Пример:

https://gd-workspace.ru
http://auth-test.ru

Для сохранения нового клиента или редактирования имеющегося, нужно нажать кнопку **«Сохранить»**. В случае успешного сохранения будет выполнена переадресация на списочную форму.

5.2.2. Ресурсы

Выбрать подраздел ***«Системные настройки» → «Ресурсы»***.

Откроется списочная форма Ресурсов.

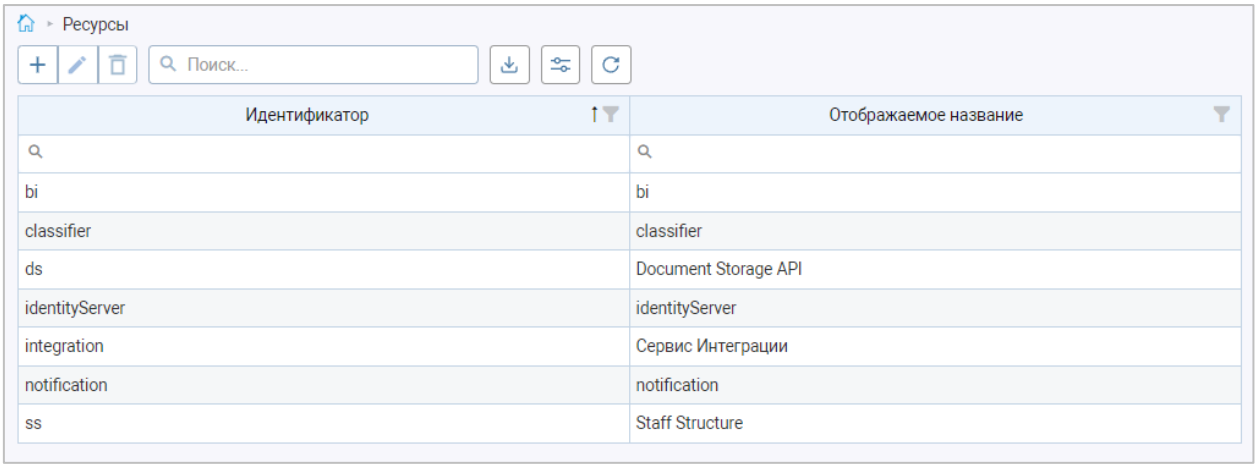


Рис. 5.6 – Ресурсы

Списочная форма Ресурсов содержит следующий набор сведений:

Идентификатор — уникальный строковый идентификатор ресурса;

Отображаемое название — читабельное представление (описание) приложения.

5.2.2.1. Создание нового ресурса

Нажать «+» над списочной формой клиентов. Будет открыта форма заполнения данных:

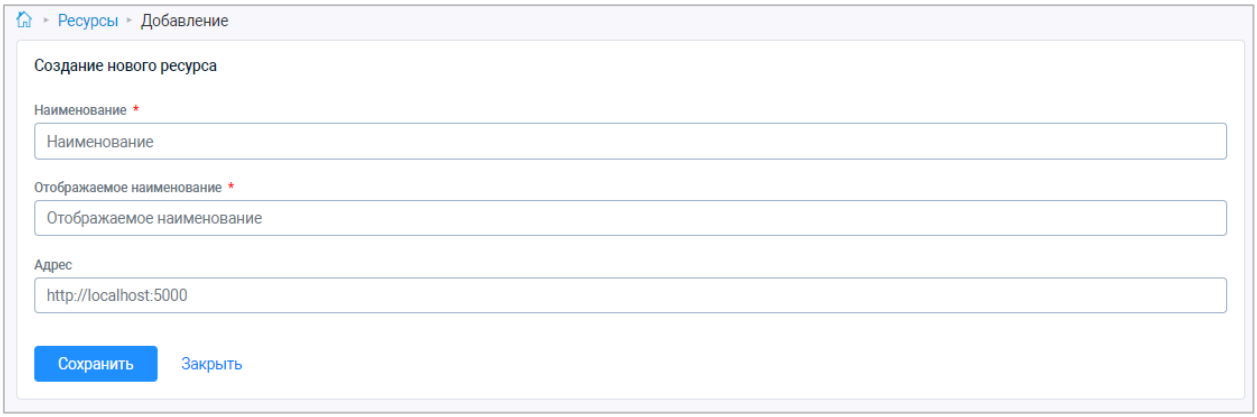


Рис. 5.7 – Создание нового ресурса

Из особенностей полей:

Наименование — не должно содержать пробелов в названии.

Адрес — должен соответствовать формату ***<схема>://[//<хост>[:<порт>]]***.

Например***: https://gd-workspace.ru***.

В случае успешного сохранения будет выполнена переадресация на карточку Ресурса.

Пример:

Рис. 5.8 – Описание ресурса

5.2.2.2. Раздел API

Если поле ***Адрес*** было заполнено корректно и Ресурс активен (запущен на сервере и имеет действующий ключ аутентификации), в данном режиме будет отражаться перечень его ***URL***, которые могут быть использованы приложениями для обращения к данным в соответствии с их бизнес-логикой.

Кнопка ***«Перезагрузить»*** позволяет вручную запросить данный список у Ресурса. Полезен для случаев, когда спецификации Ресурса изменяются и эти изменения не были занесены в сервисе Хаб во время запуска приложения.

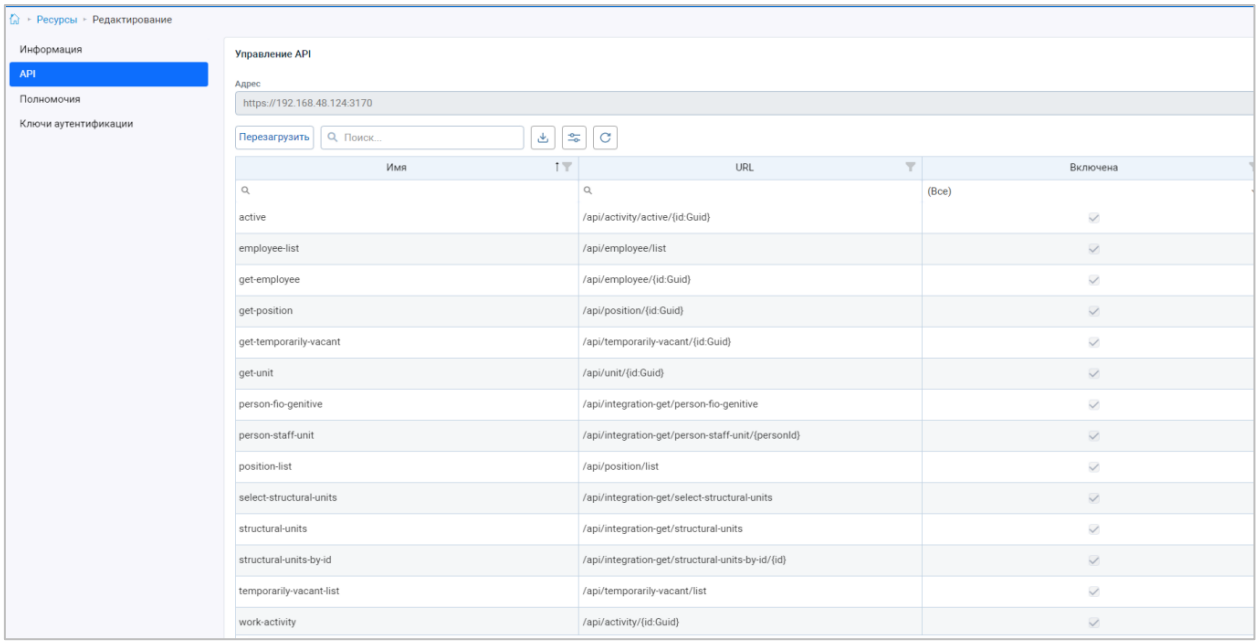


Рис. 5.9 – Управление API

5.2.2.3. Раздел «Полномочия»

В данном разделе описывается справочник прав доступа, которые использует Ресурс при обращении к своему API. Эти данные обновляются при каждом запуске Ресурса на сервере.

Носит уведомительный характер и используется как источник данных при назначении Полномочий ролям в режиме **«Роли»**.

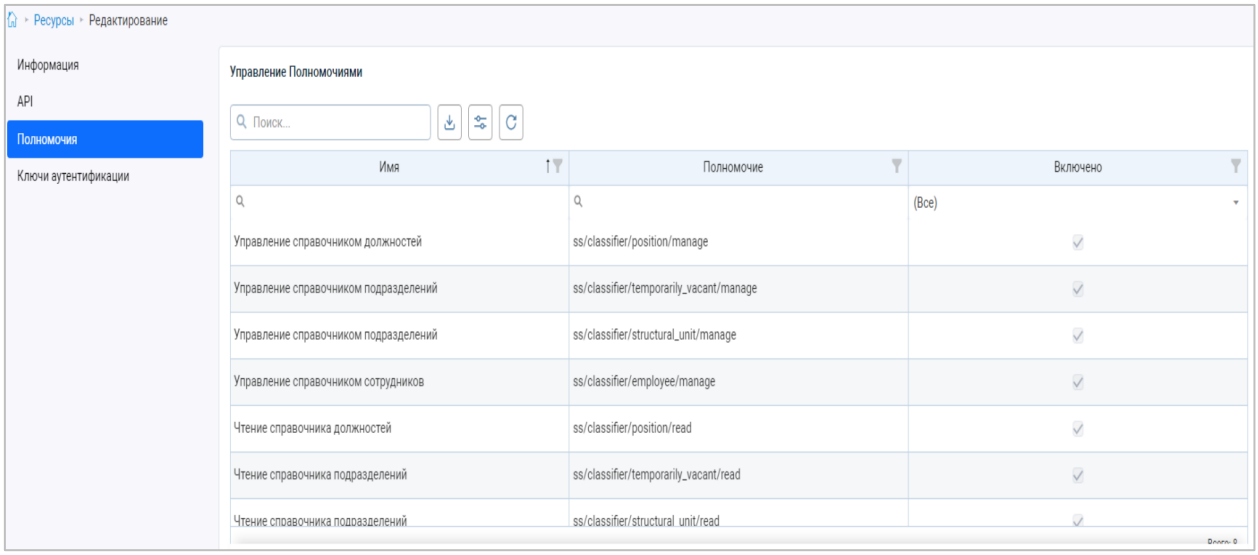


Рис. 5.10 – Управление Полномочиями

5.2.2.4. Раздел «Ключи аутентификации»

В данном разделе вносятся ключи, которые могут использоваться Ресурсом при следующих обращениях к сервису Хаб:

1) Верификация токена, полученного от Клиента;
2) Межсерверная передача данных (Полномочия);
3) Логирование и пр.

Ресурс может иметь больше одного действующего ключа, это позволяет выполнять постепенный rollout-ключей в случае кластерного развертывания Ресурса (полезно для сервисов с высокой нагрузкой).

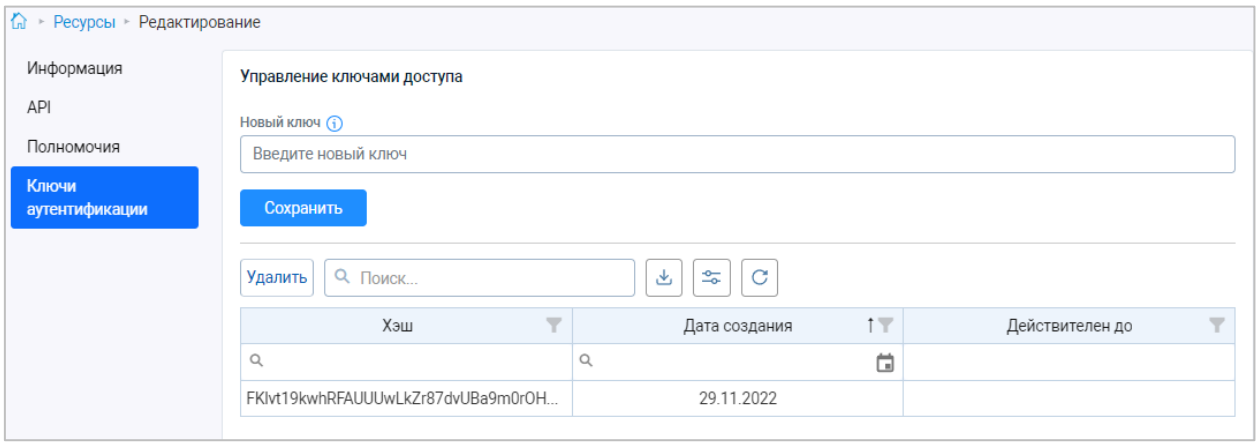


Рис. 5.11 – Управление ключами доступа

Ключи хранятся в захешированном виде, что исключает возможность их чтения после записи.

В случае отсутствия хоть одного действующего ключа, Ресурс не сможет аутентифицироваться в сервисе Хаб. Это приведет к тому, что Ресурс не сможет проверить входящие токены и будет обязан отвергнуть запросы пользователей, как недоверенные.

После настройки Администратором Системы комбинации Клиента + Ресурса, пользователь сможет зайти в Приложение (aka Клиент) по адресу на котором он развернут:

Например: ***https://192.168.48.124:6200***.

В случае успешной настройки, пользователь будет переадресован на страницу ввода Логина/Пароля, а после — обратно в приложение.

В случае если настройка была выполнена с ошибкой, то возможны следующие сценарии:

1) Ошибка настройки Клиента — будет выведена ошибка при переадресации на страницу ввода логина при входе в систему;
2) Ошибка настройки Ресурса — будет выведена **Ошибка Аутентификации** при попытке пользователя запросить данные из Ресурса,

или ошибка **Отказано в Доступе**, если у пользователя недостаточно прав для выбранного действия.

5.2.3. Роли

При описании роли:

- определяется список доступных объектов и, для некоторых объектов, права доступа к объекту (управление или только чтение);
- могут быть ограничены права на уровне записи (разделение права для пользователей в разрезе динамически меняющихся данных).

В дереве присутствуют различные объекты: объекты структуры системы (подсистемы и модули, режимы), объекты интерфейса (разделы, пункты меню), объекты с общим назначением (приказы) и другие.

Чекбокс объекта имеет вид  , если роли предоставлен доступ к объекту и ко всем подчиненным объектам текущего объекта, вид  , если предоставлен доступ к некоторым подчиненным объектам.

Чтобы предоставить доступ к объекту, следует выполнить клик на чекбоксе. При предоставлении доступа к объекту автоматически предоставляется доступ к подчиненным объектам.

Роль может быть заблокирована / разблокирована (в форме списка ролей).

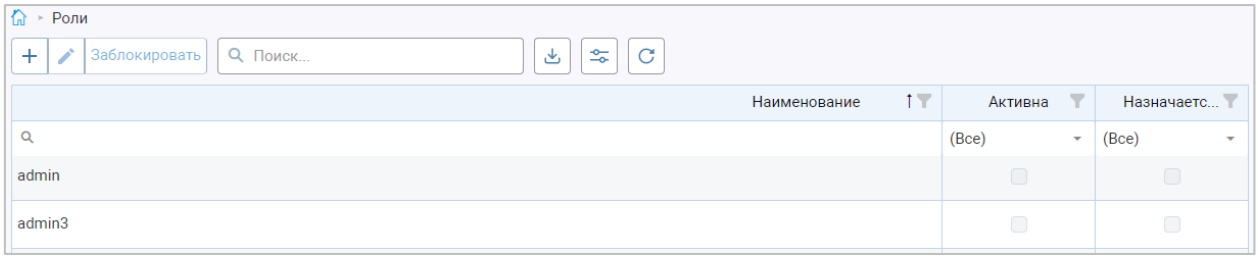


Рис. 5.12 – Роли

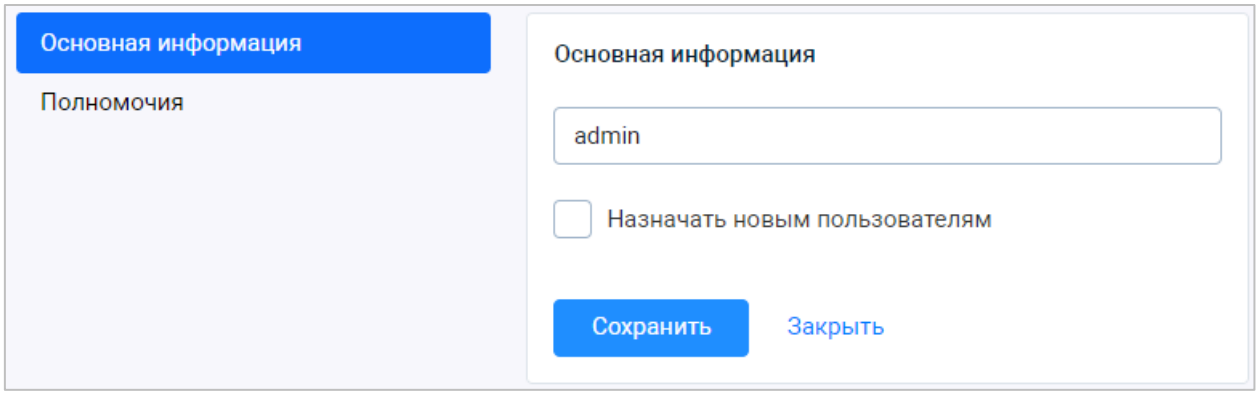


Рис. 5.13 – Роль. Основная информация

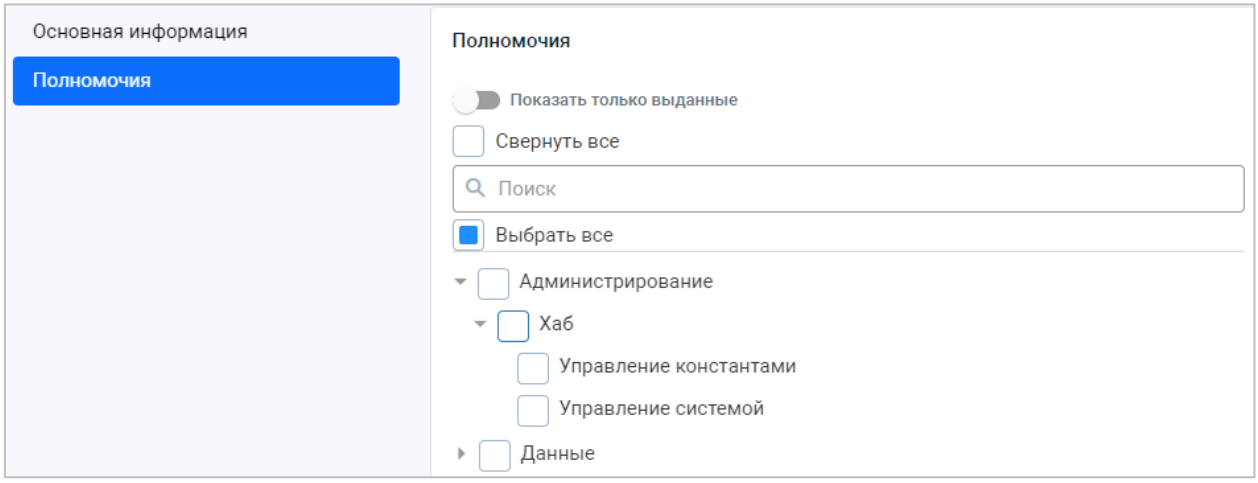


Рис. 5.14 – Роль. Полномочия

5.2.4. Пользователи

При описании пользователя определяются параметры авторизации и аутентификации (логин, пароль) и список ролей и рабочих мест (организаций/подразделений), доступных пользователю в данной роли.

Доступные организации и указанные роли формируют списки выбора пользователя (на панели управления) при работе с системой. Текущая

организация ограничивает отображение и выбор данной организацией и подчиненными подразделениями.

Пользователь может быть заблокирован / разблокирован (в форме списка пользователей).

При добавлении нового пользователя выполняется автозаполнение логина согласно настройкам генерации логина.

Доступно добавление пользователей списком – посредством импорта из файла заданного шаблона и формата.

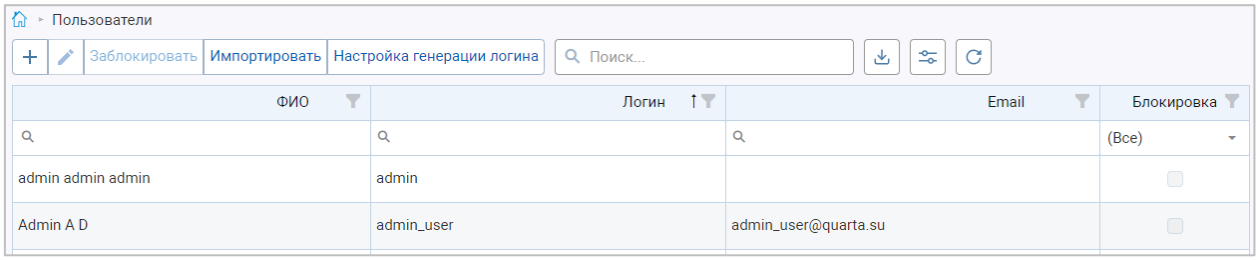


Рис. 5.15 – Пользователи

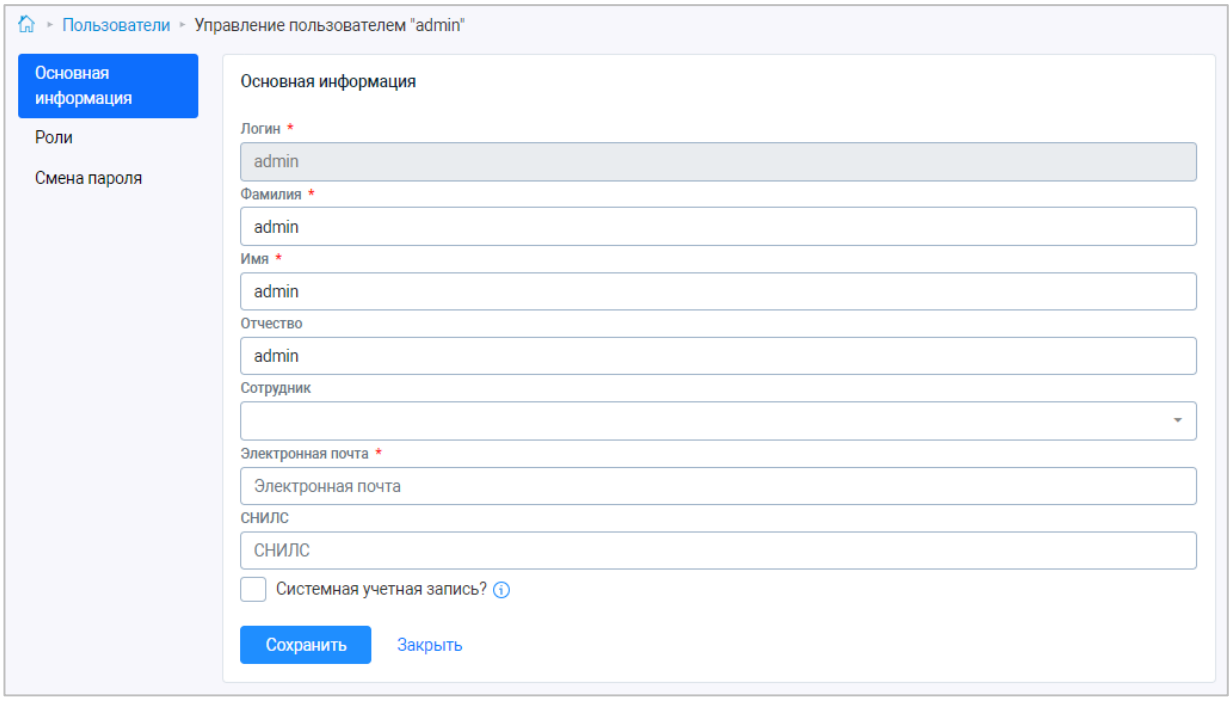


Рис. 5.16 – Пользователь. Основные сведения

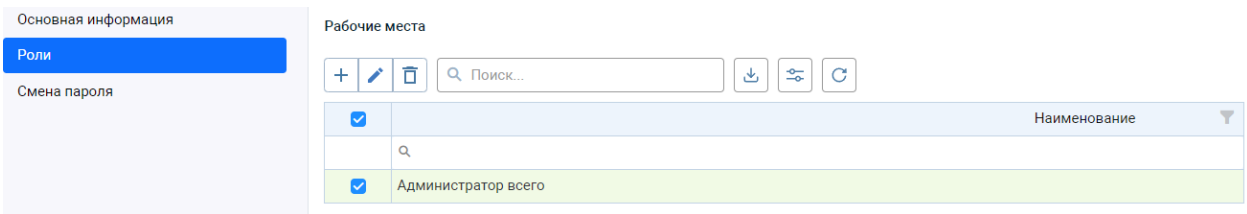


Рис. 5.17 – Пользователь. Роли

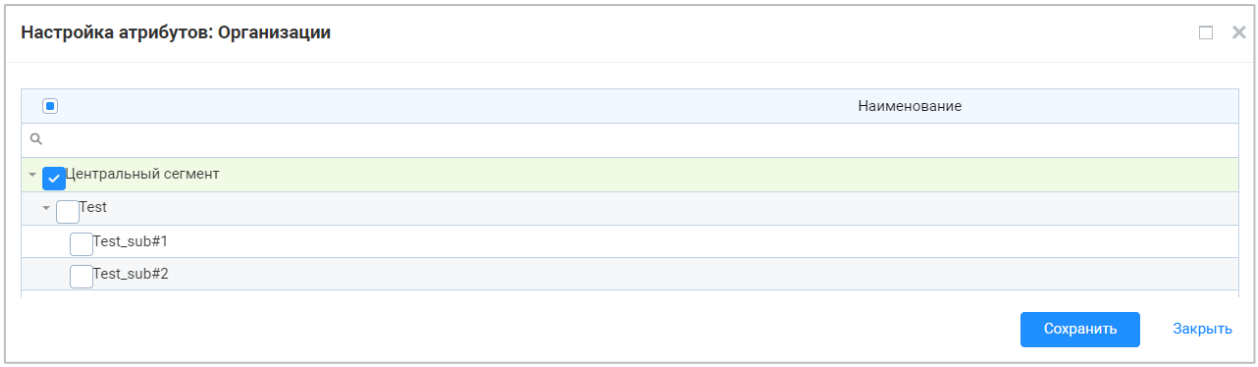


Рис. 5.18 – Пользователь. Роль. Список рабочих мест

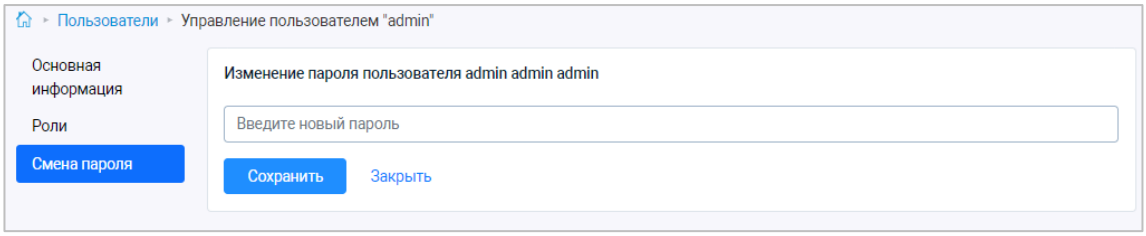


Рис. 5.19 – Пользователь. Смена пароля

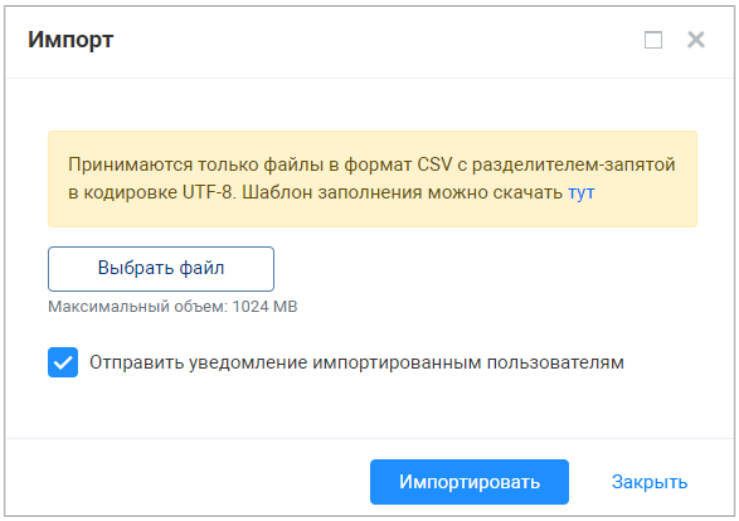


Рис. 5.20 – Импорт пользователей

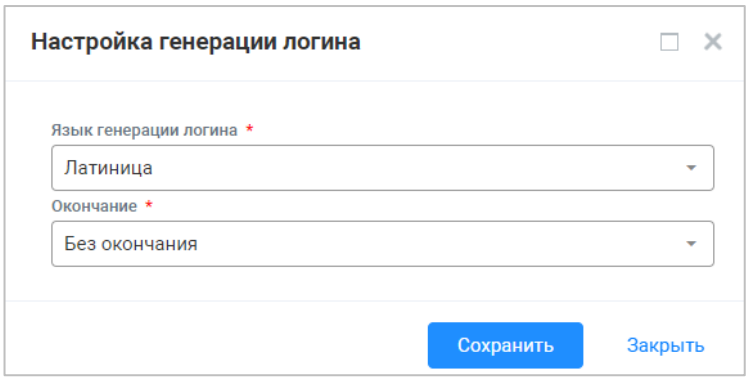


Рис. 5.21 – Настройка генерации логина пользователей

5.2.5. Сотрудники

Режим сотрудники используется для просмотра информации личных данных сотрудников.

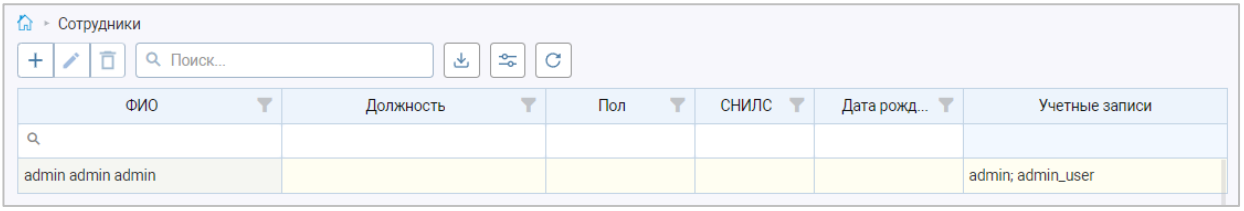


Рис. 5.22 – Сотрудники

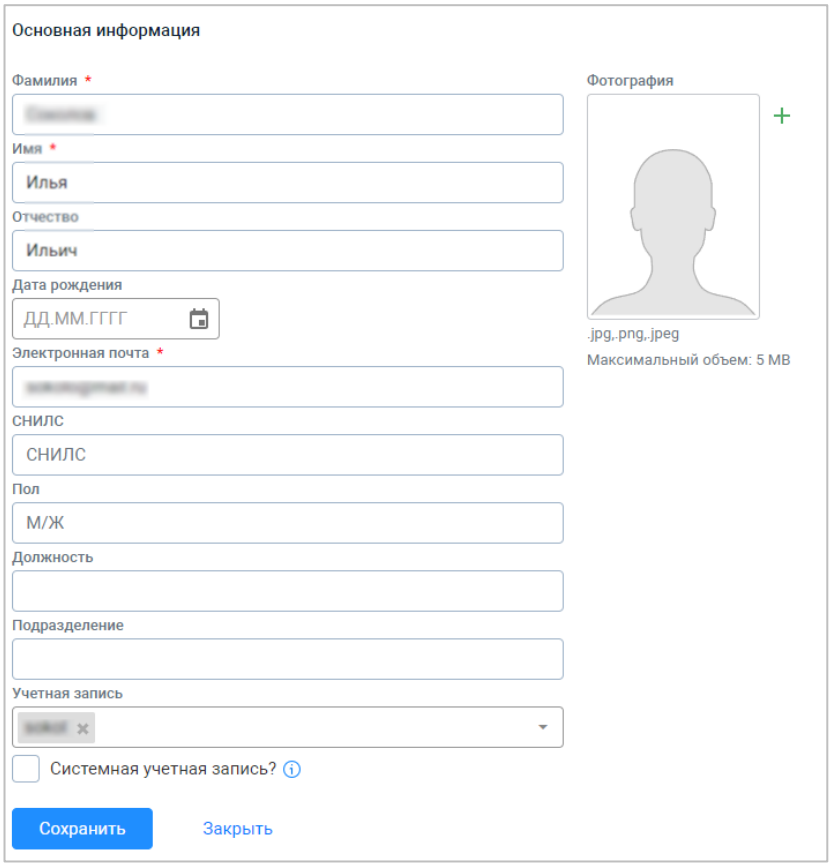


Рис. 5.23 – Сотрудники. Основная информация

5.2.6. События

Режимы просмотра событий используются для контроля подключения пользователей к системе и проверки прав доступа к объектам системы.

Справка об авторизациях пользователей (пункт меню **Активность пользователей**) представляет собой xls-отчет с общим количеством успешных и неуспешных входов пользователя под учетной записью за заданный период.

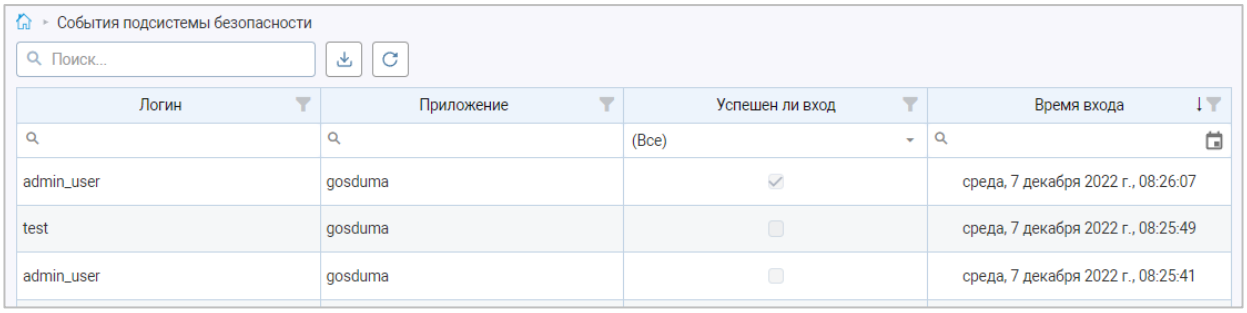


Рис. 5.24 – События аутентификации

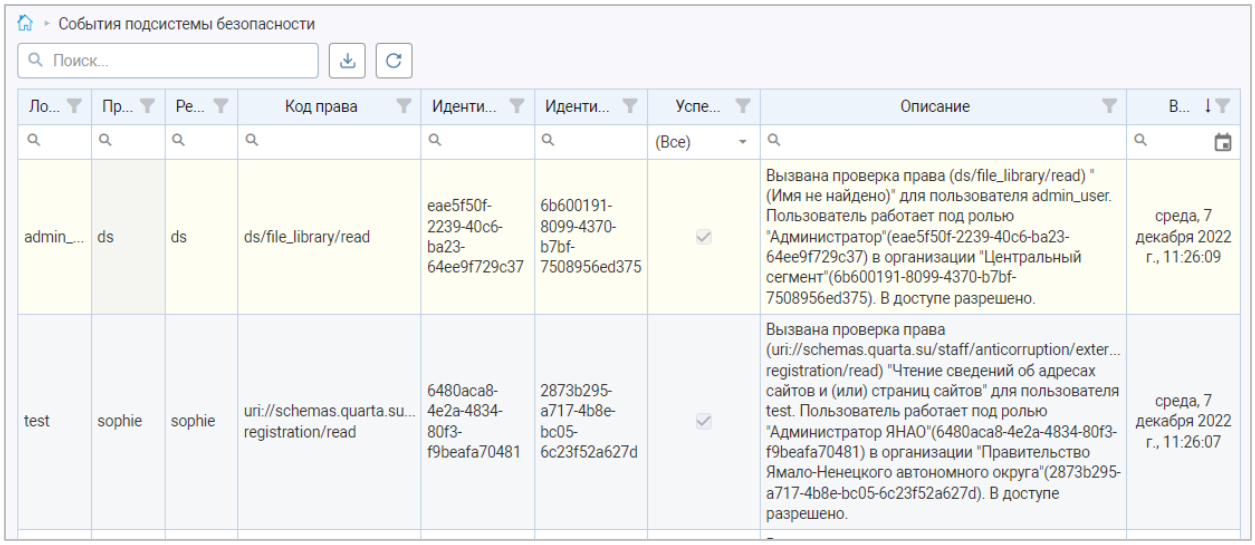


Рис. 5.25 – События безопасности

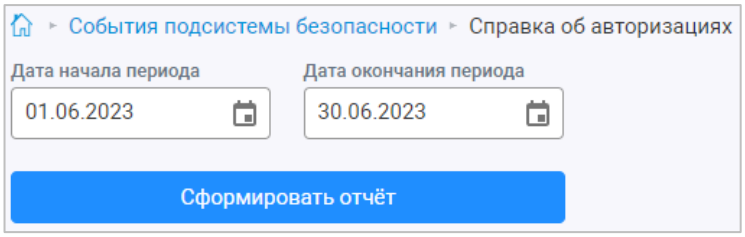


Рис. 5.26 – Настройка периода отчета

Справка об авторизациях

с 01.06.2023 по 30.06.2023

Субъект доступа	Учётная запись	Количество успешных входов	Неуспешные попытки входа
admin a. a.	admin	0	0
Admin A. D.	admin_user	62	17
Test T. T.	Test	87	1
Test T. T.	test	1	6

Рис. 5.27 – Пример отчета

5.3. Константы

Константы обеспечивают возможность настройки для организаций группы компаний различных значений параметров работы Системы.

Перечень параметров работы Системы определяется организацией-разработчиком. Пользователю доступно изменение набора доступных к выбору значений параметра и настройка значений параметра для организаций группы компаний.

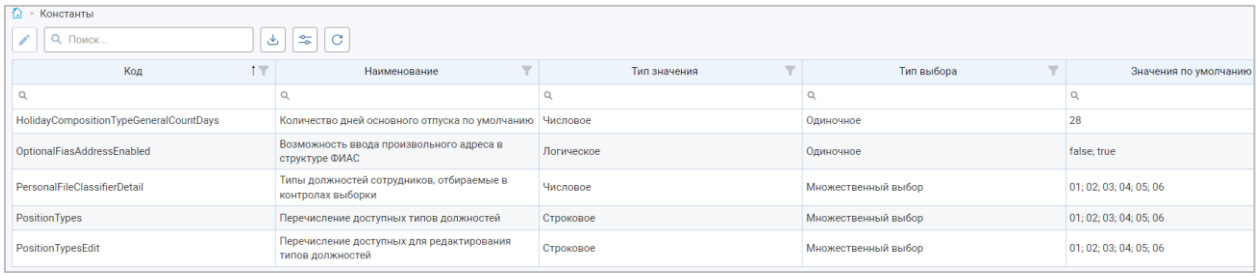


Рис. 5.28 – Список параметров

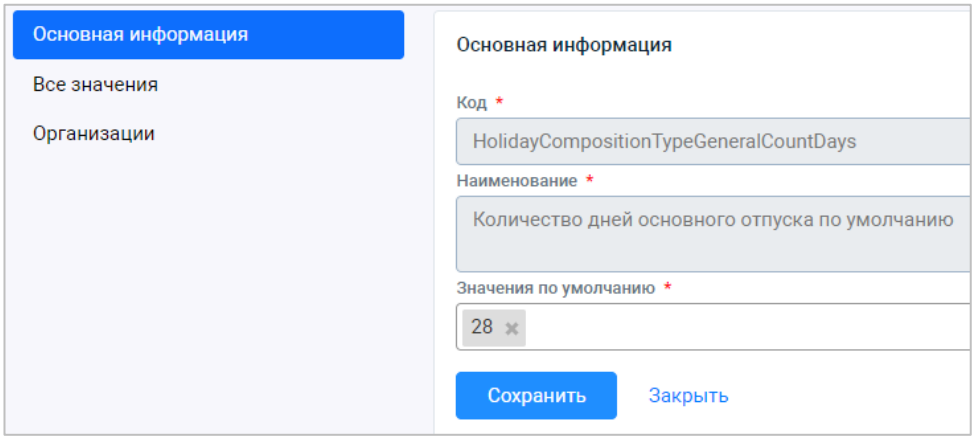


Рис. 5.29 – Параметр. Основная информация

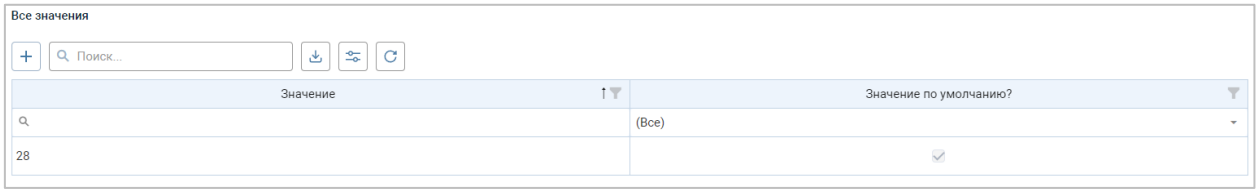


Рис. 5.30 – Параметр. Список значений

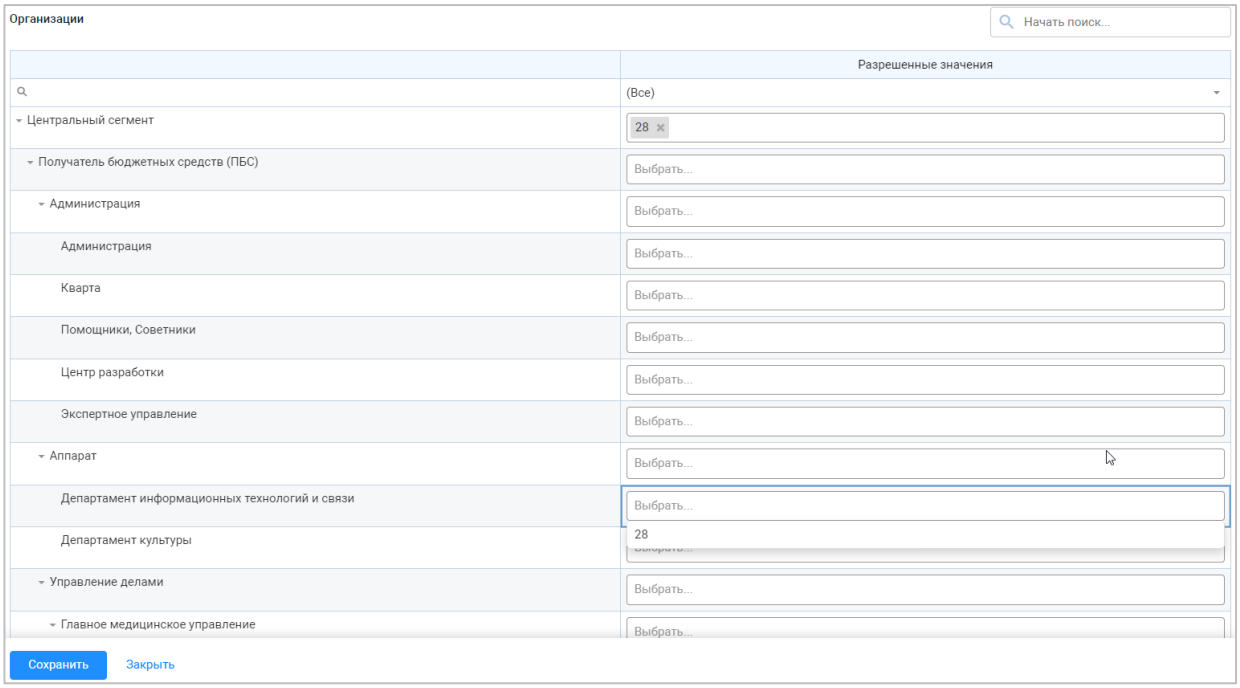


Рис. 5.31 – Параметр. Настройка значений для организаций

6. Аварийные ситуации

Отсутствие доступа к Системе, нарушение функционирования Системы, нештатное поведение Системы может возникать по причине сбоев в функционировании аппаратного обеспечения и каналов передачи данных, прикладного и специального программного обеспечения, ошибок ввода данных авторизации и ошибок администрирования Системы, а также по причине несанкционированного вмешательства в данные Системы.

В зависимости от предполагаемой причины сбоя следует проверить:

- подключение аппаратных средств и работоспособность каналов передачи данных;
- работоспособность серверного программного обеспечения и БД Системы (пп. 4.2, 4.3);
- журнал событий и корректность распределения прав доступа пользователей (п. 5), отсутствие вредоносного ПО на сервере и рабочих станциях пользователей.

Восстановление работоспособности программного обеспечения и БД Системы описано в пп. 4.4, 4.5.