«FINIST-GBANK»
ПЛАТФОРМА ДЛЯ АВТОМАТИЗАЦИИ БИЗНЕС ПРОЦЕССОВ

Казань 2024

Оглавление

1 Архитектура решения

Настоящий документ предназначен для отражения списка функциональности доступной администратору компоненты Finist-gBank (далее – Система).

Система представляет собой набор компонент, объединенных в трехзвенную модель, частями которой являются:

- Сервер приложений - служба, в которой заложена большая часть реализации обработки запросов пользователей, а так же связь с сервером хранения Баз Данных.
- Сервер базы данных - обеспечивает хранение данных.
- Тонкий клиент – работа в браузере

В простейшей конфигурации сервер приложений может быть совмещен с сервером базы данных на одной виртуальной или физической машине, к которой возможно подключение по сети одного или нескольких терминалов.

С точки зрения безопасности, надежности, масштабирования системы рекомендуется разделить сервер приложений и сервер хранения баз данных.

Система так же поддерживает установку нескольких серверов приложений для распределения нагрузки между пользователями системы.

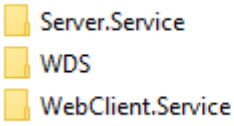
Для работы службы должны быть установлены:

  - CentOS версии 7,8 или Debian версий 9, 10 или Ubuntu версии 16.04(LTS), 18.04 (LTS), 20.04(LTS), 21.04
  - Nginx версия не ниже 1.18
  - .Net 6.0

- PostgreSQL - рекомендуется устанавливать только для ОС на базе Linux:
- Браузер(тонкий клиент) на базе платформы Chromium;

2 Поставка

В обычной поставке участвуют три архива:



Server.Service (далее - сервер): Сервер приложения. WebClient.Service (далее – веб-приложение): Сервер-приложение для тонкого клиента и Клиент-приложение для сервера. WDS (далее – тонкий клиент): Файлы тонкого клиента.

3 Предварительная настройка

Необходимо настроить учётные записи, под которыми будет осуществляться работа веб-приложения.

Учётная запись должна обладать правами уровня «Административный» После создания пользователя – перезапустить систему безопасности (Сервис – Перезапуск системы безопасности).

4 Server.Service

Конфигурируется двумя файлами:

- ***appsettings.json:*** настройки для работы веб-приложения. Необходим для разворачивания тонкого клиента.
- ***GBank.Server.Service60.dll.config***: набор настроек для сервер-приложения. В нём указываются строки подключения к БД, настройки для LDAP сервера, почтового сервера. Общение с веб-приложением происходит посредством gRPC-запросов.

4.1 GBank.Server.Service60.dll.config

4.2 appsettings.json

```
{
    "Logging": {
        "LogLevel": {
            "Default": "Information",
            "Microsoft": "Warning",
            "Microsoft.Hosting.Lifetime": "Information"
        }
    },
    "AllowedHosts": "*",
    "Kestrel": {
        "EndpointDefaults": {
            "Protocols": "Http2"
        },
        "Endpoints": {
            "Http": {
                "Url": "http://0.0.0.0:5700"
            }//,
            //"Https": {
            //  "Url": "https://0.0.0.0:5701",
            //  // Uncomment certificate section to use specified certificate.
            //  "Certificate": {
            //      "Path": "GrpcCert1.pfx",
            //      "Password": "grpc1"
            //  }
            //}
        }
        "ServerOptions": { // G.Hosting options
                "Limits": {
                        "MaxRequestBodySize": 157286400
                }
        }
    }
}
```

Секция ***AllowedHosts***:

Указываются адреса, с которых сервер принимает запросы.

Секция ***Kestrel***:

EndpointDefaults – системная секция. Версия протокола общения gRPC.

ServerOptions / Limits / MaxRequestBodySize – Максимальный размер тела запроса, в байтах.

4.2.1 HTTP/HTTPS

В секции Endpoints находятся настройки выставленных адресов для gRPC. Url – адрес, по которому будет осуществляться общение с сервером. По умолчанию мы поставляем тестовый сертификат в каталоге ***CommonFiles*** сервера.

- **Http** – вариант адреса без сертификата.
- **Https** – вариант адреса с сертификатом. Настройки сертификата указываются в секции Certificate. В секции Certificate указывается путь (Параметр Path) до pfx-сертификата и пароль от сертификата (Параметр Password).

4.3 LDAP

В серверном конфигурационном файле (GBank.Server.Service60.dll.config) есть возможность настроить LDAP сервер, для аутентификации пользователей через доменные учётные данные. LDAP серверов может быть несколько. В секции G.Server необходимо настроить строку подключения к серверу БД PostgreSQL:

```
<ConnectionString value="Server=СЕРВЕР БД;Port=5432;Database=Название БД;User ID=ЛОГИН;Password=ПАРОЛЬ;Include Error Detail=true;SSL Mode=Require;Trust Server Certificate=true" />
```

```
<Ldap>
    <!--
                key - уникальный ключ сервера; обязательный параметр; сервер с ключом some_server будет обрабатывать аутентификацию пользователей с
                    именами вида some_server\имя_пользователя; обязательный параметр
                host - адрес сервера LDAP; обязательный параметр
                port - порт для подключения; обязательный параметр
                useSsl - флаг использования SSL; по умолчанию false
                trustServerCertificate - флаг отключения проверки серверного сертификата; работает только при useSsl="true"; по умолчанию false
                repeatedAuthenticationInterval - интервал повторной аутентификации пользователей; формат - System.TimeStamp; не может превышать 1 день; по умолчанию 10 минут
                authenticationType - тип аутентификации; значение соответствует enum'y System.DirectoryServices.Protocols.AuthType; по умолчанию Basic
                protocolVersion - версия протокола LDAP; по умолчанию 3
                displayName - отображаемое имя сервера
            -->
    <Server key="some_server" host="some_server.com" port="389" useSsl="false" trustServerCertificate="false" repeatedAuthenticationInterval="0:10:0"
                authenticationType="Basic" protocolVersion="3" displayName="Some server">
        <GAuthentication>
            <!-- Настройки операции LDAP bind (первоначальная аутентификация, используется для поиска аутентифицированного пользователя) -->
            <!-- dynamic - флаг использования учетной записи текущего пользователя для первоначальной аутентификации; обязательный параметр
                 dn - distinguished name учетной записи пользователя, использующейся для первоначальной аутентификации; если dynamic="true", то может содержать плейсхолдер
                    логина текущего пользователя %%login%%; для пользователя some_server\some_user %%login%% == some_user; обязательный параметр
                 password - пароль учетной записи пользователя, использующейся для первоначальной аутентификации; используется только если dynamic="false"; необязательный параметр -->
            <Bind dynamic="true" dn="uid=%%login%%,ou=Users,dc=some_server,dc=com" password="some_password" />
            <!-- Настройки LDAP query, использующегося для поиска аутентифицированного пользователя -->
            <!-- root - distinguished name каталога, с которого начинается поиск; обязательный параметр
```

```
                filter - LDAP запрос для поиска пользователя; д.б. составлен так чтобы не вернуть больше одной записи
(нужен uid/sAMAccountName); не должен

                возвращать заблокированных пользователей (критерии нужно согласовать с администратором сервера);
обязательный параметр -->

            <Search root="dc=some_server,dc=com" filter="(&amp;(uid=%%login%%)(objectClass=posixAccount))" />

        </GAuthentication>

    </Server>

    <!-- Пример настройки сервера для аутентификации в ActiveDirectory (на базе AD esterdev) -->

    <Server key="esterdev" host="bighead.esterdev.com" port="389" useSsl="false" trustServerCertificate="false"
repeatedAuthenticationInterval="0:10:0" authenticationType="Basic" protocolVersion="3" displayName="EsterDev (LDAP)">

        <GAuthentication>

            <Bind dynamic="true" dn="%%login%%@esterdev.com" />

            <Search                                                                    root="dc=esterdev,dc=com"
filter="(&amp;(sAMAccountName=%%login%%)(objectCategory=person)(objectClass=user)(!(userAccountControl:1.2.840.11355
6.1.4.803:=2)))" />

        </GAuthentication>

    </Server>

    <!-- Пример настройки сервера для аутентификации в openLDAP (на базе локальной тестовой платформы) -->

    <Server   key="ldaptest"   host="192.168.1.114"   port="389"   useSsl="false"   trustServerCertificate="false"
repeatedAuthenticationInterval="0:10:0" authenticationType="Basic" protocolVersion="3">

        <GAuthentication>

            <Bind dynamic="true" dn="uid=%%login%%,ou=Users,dc=ldaptest,dc=com" />

            <Search root="dc=ldaptest,dc=com" filter="(&amp;(uid=%%login%%)(objectClass=posixAccount))" />

        </GAuthentication>

    </Server>

</Ldap>
```

5 *GBaseSolution*.WebClient

Конфигурируется двумя файлами:

- ***GBank.WebClient.Service.dll.config*** – основной файл конфигурации. Используется для настройки общения веб-приложения с сервером.
- ***appsettings.json*** – файл конфигурации для настройки общения веб-приложения с тонким клиентом.

5.1 *GBank.WebClient.Service.dll.config*

```
...
<G.Client>
    ...
```

```
    <Credential type="Basic" login="1" password="1" profile="Базовый профиль"/>
</G.Client>
<G.Common>
    <HostAddress value="http://localhost:5700" />
</G.Common>
...
```

В данном файле нас интересует секция **G.Client** и **G.Commo**n.

HostAddress – адрес, который мы выставили в appsettings.json сервера. По нему будет осуществляться общение веб-приложение – сервер.
Credential – информация для авторизации веб-приложения на сервере. Type basic – не меняем, способ авторизации через связку логин/пароль. Login/password – данные пользователя **внутри системы опер.рисков**.

Пользователь должен обладать административным уровнем доступа для корректной работы.

Profile – наименование профиля пользователя в системе.

5.2 appsettings.json

```
{
    "AllowedHosts": "*",
    "Kestrel": {
        "Endpoints": {
            "Http": {
                "Url": "http://localhost:5050"
            },
            "Https": {
                "Url": "https://localhost:5051"
                // Uncomment following section for use specified certificate for API hosting.
                //"Certificate": {
                //    "Path": "CommonFiles/DevRiskCert.pfx",
                //    "Password": "DevRisk"
                //}
            }
        }
    },
    "CPO": {
        "Names": ["http://192.168.8.53", "https://192.168.8.54"]
    }
```

```
}
```

Endpoints – Настройки выставленных адресов для gRPC. Url – адрес, по которому будут приниматься запросы от тонкого клиента. Http – вариант адреса без сертификата.

Https – вариант адреса с сертификатом. Настройки сертификата указываются в секции Certificate. В секции Certificate указывается путь (Опция Path) до pfx-сертификата и пароль от сертификата (Опция Password).
CPO – (CrossPolicyOrigins) секция для настройки CORS (Cross-origin resource sharing). Она позволяет указать, с каких адресов веб-приложение будет принимать запросы.

6 WDS

Набор файлов тонкого клиента.

В этом наборе можно заменить файлы:

- favicon.ico – иконка веб-сайта в браузере.

Так же, в папке assets можно заменить/редактировать следующие файлы:

- logo.png – логотип приложения на странице входа.
- styles/global-colors.css – цвета приложения.

Файлы из архива необходимо распаковать в каталог public_html веб-сервера.

7 Настройка веб-сервера, проксирование

Тонкий клиент направляет запросы «сам на себя», с дополнительной частью пути «/api/». Таким образом, в веб-сервере (в нашем случае nginx) мы должны настроить проксирование по пути */api/*

```
location /api/
{
   proxy_pass https://адрес-webclient:webclient-порт$request_uri;
   proxy_redirect off;
}
```

7.1 Настройка веб-сервера, HTTPS

Для подписания веб-сайта сертификатом потребуется связка публичный сертификат и секретный ключ.

```
ssl_certificate     www.example.com.crt;
ssl_certificate_key www.example.com.key;
```

Секретный ключ (ssl_certificate_key) необходимо хранить в файле с ограниченным доступом, но Nginx'y необходимы права для чтения на этот файл.

8 Регистрация сервисов

Необходимо отредактировать файлы сервисов под окружение:

```
[Unit]
Description=CRM Web Client Service

[Service]
Type=notify
# will set the Current Working Directory (CWD). Worker service will have issues without this
setting
WorkingDirectory=/opt/EM/CRM.WebClient.Service/

# systemd will run this executable to start the service
ExecStart=/usr/bin/dotnet /opt/EM/CRM.WebClient.Service/CRM.WebClient.Service.dll &

# to query logs using journalctl, set a logical name here
SyslogIdentifier= CRM.WebClient.Service

# Use your username to keep things simple.
# If you pick a different user, make sure dotnet and all permissions are set correctly to
run the app
# To update permissions, use 'chown yourusername -R /opt/EM/CRM.WebClient.Service' to take
ownership of the folder and files,
#     Use 'chmod +x /opt/EM/CRM.WebClient.Service/CRM.WebClient.Service.dll' to allow
execution of the executable file
User=adm

# ensure the service restarts after crashing
# Restart=always
# amount of time to wait before restarting the service
# RestartSec=15

# This environment variable is necessary when dotnet isn't loaded for the specified user.
# To figure out this value, run 'env | grep DOTNET_ROOT' when dotnet has been loaded into
your shell.
# Environment=DOTNET_ROOT=/opt/rh/rh-dotnet31/root/usr/lib64/dotnet
# Environment=DOTNET_ROOT=/usr/bin/dotnet
```

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

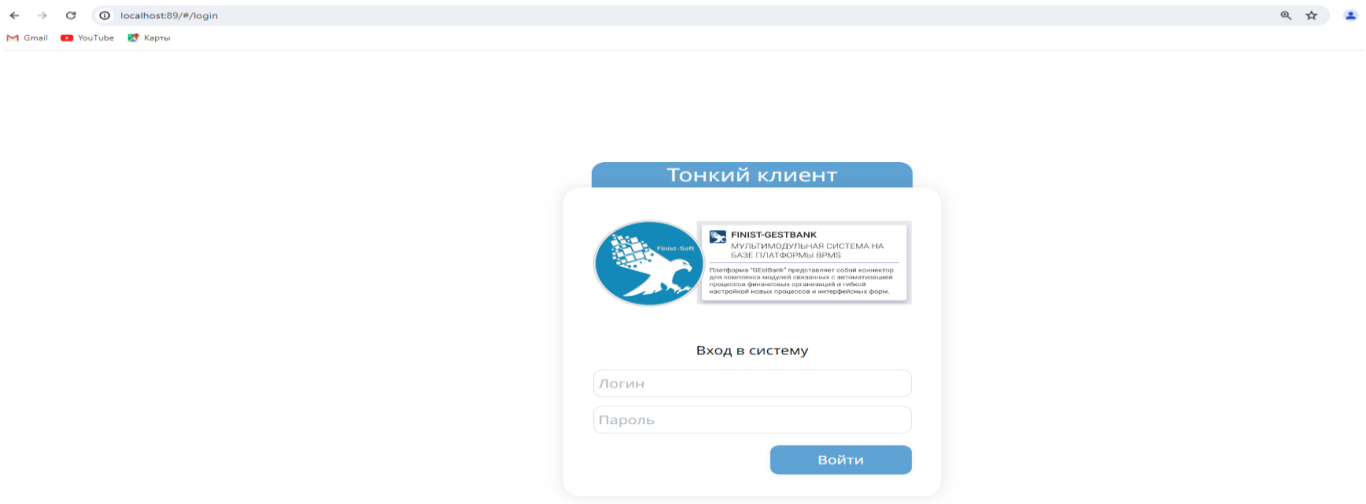
Основные конфигурационные поля: WorkingDirectory - каталог содержащий файлы и конфиги сервиса. ExecStart – Исполняемая команда при запуске сервиса через systemctl. Указываем путь до исполняемой dll. Конфигурационный файл применим как к серверу, так и к веб-клиенту, необходимо лишь заменить пути до запускаемых файлов. User – Пользователь, под которым будет осуществлён запуск и исполнение приложения.

Файлы сервисов подкладываем в /etc/systemd/system, после чего нам доступен запуск через **sudo systemctl start Server.Worker.service**

9 Проверка работоспособности

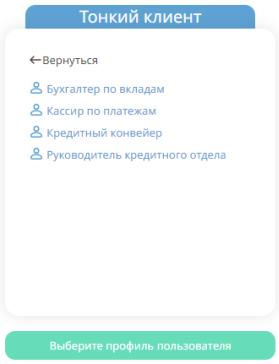
Если ошибок не возникло – можно воспользоваться приложением. Открыть веб-браузер, ввести адрес машины/глобальный адрес машины, на которой развёрнут веб-сервер.

При корректной настройке веб-сервера/указания адреса – откроется стандартная форма логина. Вход в приложение можно осуществить при помощи связки логин (с доменом) + пароль, от windows-аккаунта, в случае, если настроен LDAP сервер.

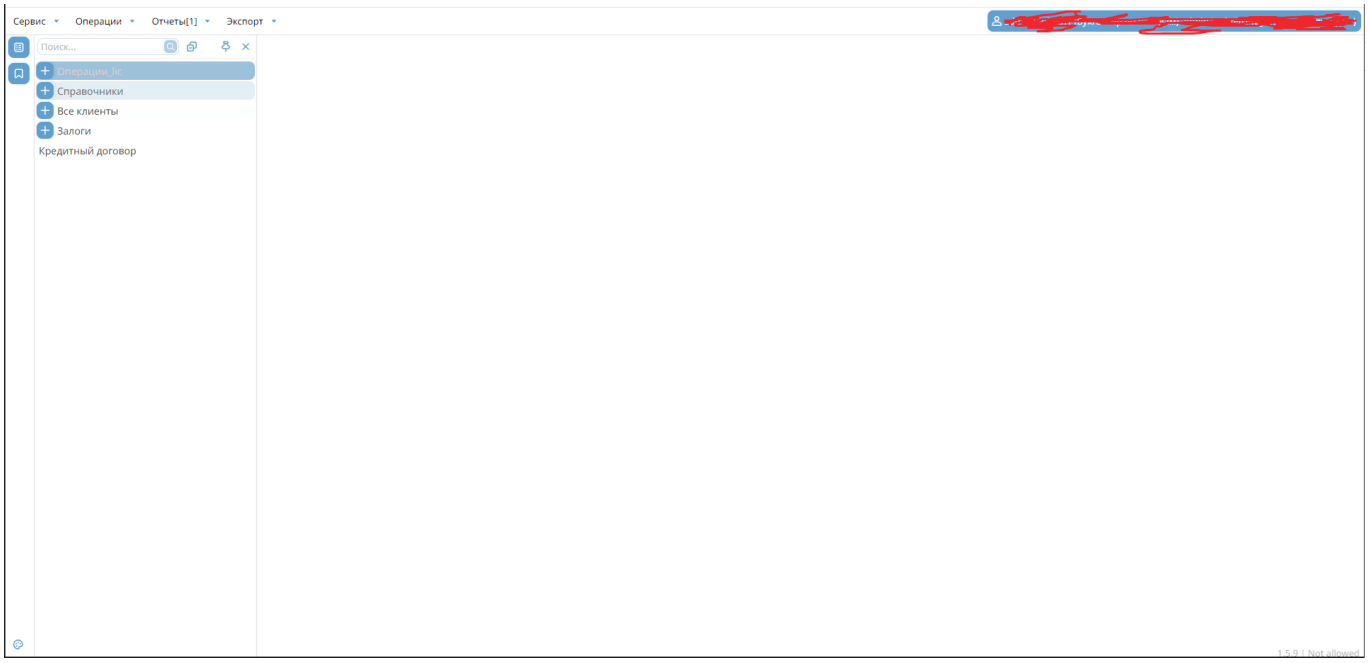


*цветовая схема и логотип могут быть изменены.

В форме указываем логин/пароль пользователя. Если всё настроено корректно, то откроется окно выбора рабочего профиля (если профилей несколько)



После выбора – нас перенесёт на основную страницу приложения:



*Выход осуществляется по нажатию иконки «пользователь», в контекстном меню будет кнопка «Выйти»

10 Проблемы, которые могут возникнуть

10.1 При авторизации по корректному логину/паролю ничего не происходит.

Если не возникло никаких дополнительных информационных сообщений, то, скорее всего, дело в том, что был добавлен новый логин для пользователя/пользователь. В таком случае, система безопасности не перезагружена и сервер-приложение не знает о новом логине пользователя.

10.2 При авторизации возникает ошибка 502 Bad gateway

Одно из возможных решений - проверить настройку веб-сервера. Корректно ли он перенаправляет запросы.
Если запросы перенаправляются корректно, то, возможно, ошибка в настройке веб-приложения. Необходимо убедиться, что в конфигурационном файле appsettings.json закомментирована секция https (при отсутствии сертификата).

10.3 При авторизации возникает ошибка 408 Request timeout

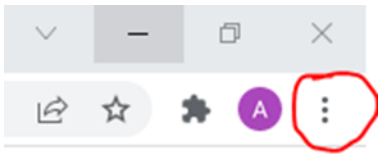
Необходимо убедиться, что веб-приложение корректно натравлено на сервер-приложение. Так же, проверить, что сервер-приложение запущено и работает.

10.4 При авторизации возникает ошибка 500 | Другие ошибки

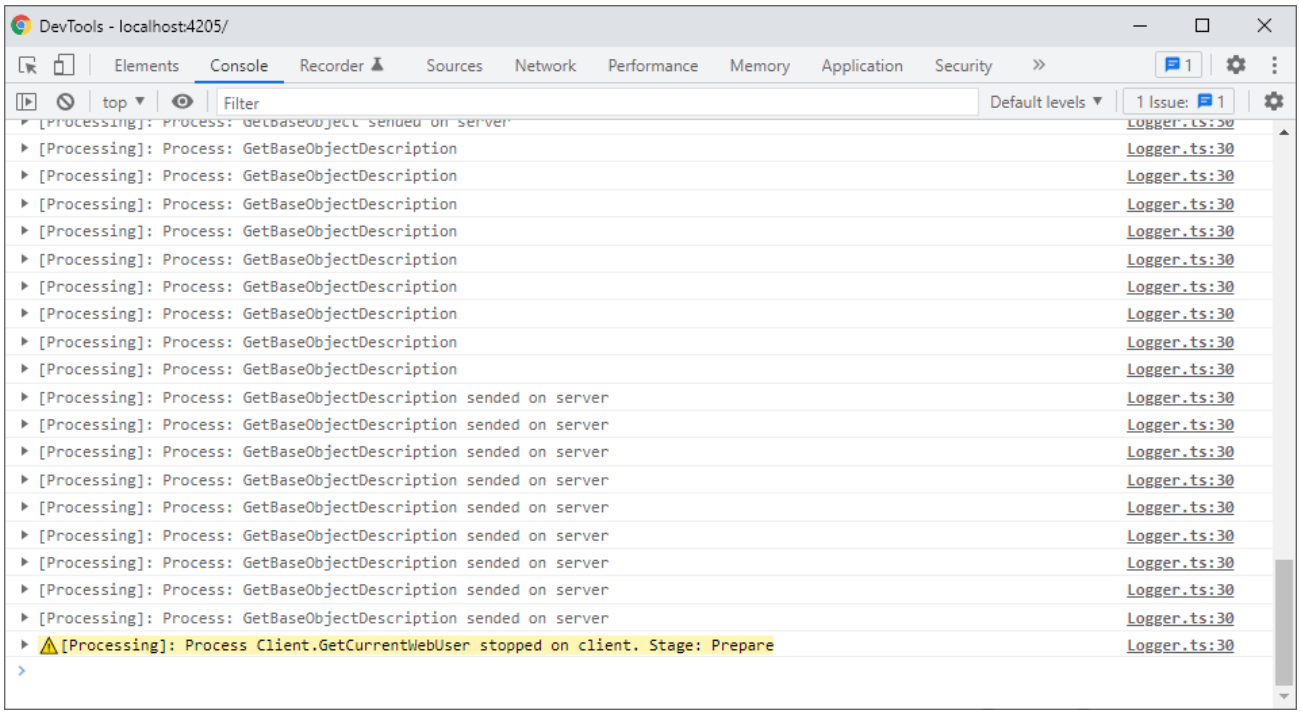
Необходимо связаться с разработчиками, приложив файл .har из консоли разработчика, секции networking и лог консоли из секции console.

Сформировать эти файлы можно таким образом:

Открываем консоль разработчика (Google Chrome) меню браузера (три точки)

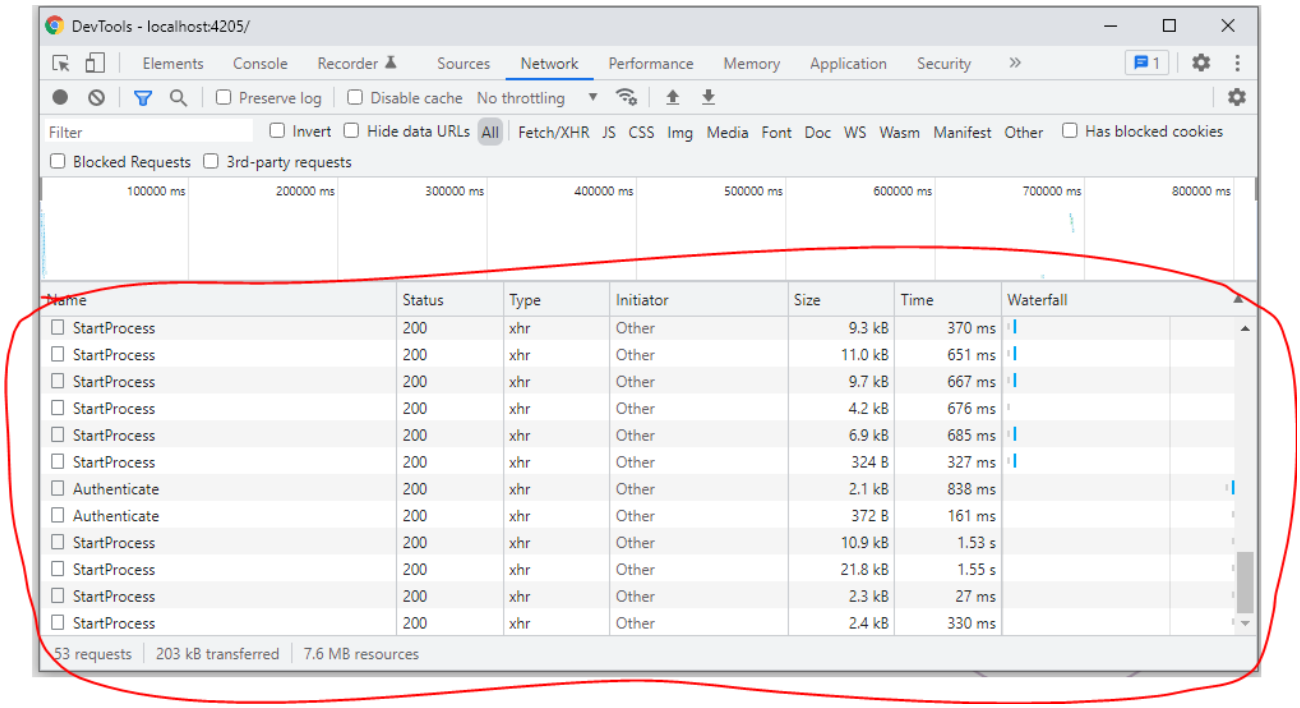


Секция меню «Больше инструментов», кнопка «Консоль разработчика» (Инструменты разработчика).
Откроется окно с инструментами:



Во вкладке «Console» (Консоль) нажимаем ПКМ – «Сохранить как». Сформируется текстовый файл, его прикладываем к письму.

Далее, переходим во вкладку «Network» (Сеть):



Нажимаем ПКМ в выделенной области, выбираем пункт меню «Сохранить всё как HAR» (Save all as HAR with content). Сформированный файл прикладываем к письму.

11 Технические требования

11.1 Требования к СУБД

Для функционирования системы на сервере требуется:

11.1.1 Программное обеспечение СУБД:

- PostgreSQL - рекомендуется устанавливать только для ОС на базе Linux:
- PostgreSQL от версии 13;
- pgAdmin III

11.1.2 Аппаратное обеспечение сервера:

- Процессор: x86-совместимый 64-разрядный (Intel, AMD); 4 ядра с частотой от 2 ГГц.
- Оперативная память: 32ГБ.
- Дисковая подсистема: от 100 ГБ.
- Канала связи: от 100 Mbit/sec, при кол-ве пользователей более 50 рекомендуется от 1000 Mbit/sec.

11.2 Требования к конфигурации сервера приложений

Для функционирования системы на сервере требуется:

11.2.1 Программное обеспечение сервера

- Linux:
  - CentOS версии 7,8
  - Debian версий 9, 10
  - Ubuntu версии 16.04(LTS), 18.04 (LTS), 20.04(LTS), 21.04
  - .NET 6.0
  - Необходимо установить пакеты «libc6-dev» и «libgdiplus»

11.2.2 Аппаратное обеспечение сервера

- Основные требования к аппаратному обеспечению вытекают из требований используемой версии ОС и СУБД.
- Процессор: x86-совместимый 64-разрядный (Intel, AMD); 4 ядра с частотой от 2 ГГц.

- Оперативная память: 16ГБ.
- Дисковая подсистема: от 50 ГБ.
- Канал связи: от 100 Mbit/sec, при кол-ве пользователей более 50 рекомендуется 1000 Mbit/sec.

11.3 Рабочая станция:

11.3.1 Программное обеспечение ОС рабочей станции Администратора:

- Должен быть установлен пакет .Net в зависимости от ОС:
  - Linux: Framework 5.0

11.3.2 Аппаратное обеспечение рабочей станции Администратора:

Основные требования к аппаратному обеспечению вытекают из требований используемой версии ОС.

- Процессор: x86-совместимый 64-разрядный (Intel, AMD).
- Оперативная память: от 8Gb
- Разрешение монитора: 1920x1080 и выше.

11.3.3 Программное обеспечение ОС рабочей станции Пользователя:

- Браузер(тонкий клиент) на базе платформы Chromium;
- Аппаратное обеспечение рабочей станции пользователя вытекает из требований работы с браузером;

Для выполнения операций, требующих больших вычислительных ресурсов, таких как импорт / экспорт данных или получение сложных отчетных форм, рекомендуется использовать более мощные рабочие станции с характеристиками выше рекомендуемых.