Поставка ПО

«Finist-Report.IFRS9»

Инструкция администратора

Казань 2024

Оглавление

1 Аннотация

Настоящий документ является руководством администратора по технической настройке системы.

2 Системные требования

2.1 Требования к клиентской машине

Для работы с системой на рабочем компьютере конечного пользователя должен быть установлен .net 6.0

2.2 Требования к серверной машине

2.2.1 Linux

2.2.1.1 Аппаратное обеспечение сервера

- Процессор x86-совместимый, 64-разрядный (AMD, Intel). 4 ядра с частотой от 2ГГц.
- Оперативная память 16Гб.
- Дисковая подсистема от 50Гб.

2.2.1.2 Программное обеспечение сервера

- Установленный пакет Net 6.0 runtime.
- Операционная система Ubuntu 20.04.5 и выше.

3 Состав поставки

В состав поставки входят следующие каталоги (папки):

- **FinistDigitalBank.Report.Server.Worker** – Сервер-приложение;
- **FinistDigitalBank.Report.UserArm** (далее – толстый клиент): файлы толстого клиента. Только для Windows.

Если ПО устанавливается впервые, то в состав поставки также входит файл базы данных (файл-бэкапа).

4 Разворачиваем приложения на сервере

4.1 Linux

Копирование файлов будем осуществлять на примере файлового менеджера *Far Manager*.

4.1.1 Навигация в Far Manager

Alt + F1 / F2 – переключение отображаемой директории в окнах, где F1 – левое, а F2 – правое.
Tab – переключение между окнами.
Стрелки на клавиатуре – перемещение в окне.
F5 – копирование файлов.
F7 – создание нового каталога в открытом окне.
F8 – удаление файлов/каталогов.
Insert – выделение файлов/каталогов.

4.1.2 Перенос файлов при помощи Far Manager

При помощи Far'а создаём новое подключение командами Alt + F1 (изменить левое окно), переключаемся на NetBox.

В открывшейся вкладке создаём новое соединение при помощи сочетания клавиш Shift + F4.
Заполняем поля полученными данными. Нажимаем Ok.
Появится предупреждение с вопросом – «*точно хотим сохранить пароль?*». Сохраняем.
Далее откроется окно, где мы можем дать удобное наименование новому подключению

В результате наших действий в списке появится новая запись.

Выделяем нашу запись и нажимаем кнопку Enter – открывается новое соединение

После успешной установки подключения в левой части экрана будут отображаться каталоги сервера, а справа – каталоги нашего компьютера.

Переходим на вкладку сервера и при помощи клавиши F7 создаём там новый каталог. Называем его «finist-soft».

При наличии ошибки «Permission denied» необходимо запросить у ответственных сотрудников банка права на создание и копирование файлов на сервер для нашего пользователя.

При помощи *навигации* в правом окне (с файлами поставки) переходим к каталогу с файлами приложения, выделяем их (клавиша Insert) и копируем их при помощи клавиши F5, нажимаем Copy.

По завершении копирования файлов поставки с нашего компьютера на сервер, переходим к *настройке базы данных*, а затем к *настройке приложений*.

4.1.3 Редактирование файлов в Far Manager

Для изменения файлов конфигурации и настройки приложений необходимо, чтобы пользователь обладал правами на редактирование файлов. Само редактирование происходит при помощи команды Edit (клавиша F4).

После внесения изменений необходимо сохранить файл - команда Save (клавиша F2).
Или при закрытии изменённого файла при помощи команды Quit (клавиша F10)

5 Настройка базы данных

5.1 PostgreSQL (для Linux-сервера)

5.1.1 DBeaver

Для этого способа необходимо, чтобы сервер базы данных уже был настроен и база-пустышка, которую будем восстанавливать, существовала.

! Если этого сделано не было, то подключение и настройка БД через DBeaver невозможна. В этом случае Воспользуйтесь настройкой по SSH

Рассмотрим процесс развёртывания базы при помощи бесплатного ПО *DBeaver Community*.

Устанавливаем и запускаем DBeaver, создаём новое подключением нажатием на «вилку», выбираем PostgreSQL двойным щелчком мыши.

Открывается окно с настройками подключения.
В данном окне необходимо ввести данные сервера базы данных (БД):
Блок Server:
Host – IP-адрес или имя хоста сервера БД.
Database – база данных, к которой мы осуществляем подключение.
Блок Authentication:
Username – имя пользователя, под которым будет осуществляться управление базой. Он должен иметь права на управление этой базой.
Password – пароль пользователя БД, под которым мы будем управлять БД.
Заполняем все вышеперечисленные поля и нажимаем «ОК».
В случае успешной настройки в навигаторе DBeaver курсивом отобразится информация о подключении. Если подключение к БД прошло успешно, возле значка подключения (postgresql) появится зелёная галка. Разворачиваем его.

Открываем каталог Databases и находим нужную нам базу данных.

Вызываем контекстное меню нажатием ПКМ по имени БД, переходим в Tools, выбираем пункт Restore.

Открывается окно восстановления базы.

Нас интересует поле «Backup file».

Прописываем полный путь к нашему файлу бэкапа БД вручную или нажимаем на значок для выбора нужного файла из проводника. В последнем случае открывается окно, где в выпадающем списке с типом файлов необходимо установить * (*) (любой тип) – это отобразит все файлы в папке.

Выбираем файл с бакапом базы данных.

Нажимаем кнопку Start для запуска процесса восставления БД из бэкапа.

Запускается процесс восстановления БД из нашего файла бэкапа. Ждём окончания процесса.

После завершения восстановлени БД приступаем к *настройке сервера.*

5.1.2 SSH

Нам потребуется клиент SSH – встроенный в операционную систему либо иной другой с поддержкой sftp и командной строки, например, *Putty*.

ПОДКЛЮЧАЕМСЯ ПО SSH К СЕРВЕРУ БД.

Рассмотрим на примере windows - открываем PowerShell и выполняем команду:
ssh **user@remoteAddress**

В случае, если банк использует специфичный порт для подключения, необходимо добавить аргумент «-P»:

ssh **-P portNumber user@remoteAddress**

**красные поля – значения, которые нужно заполнить данными от банка.*

После успешного подключения система запросит пароль.

! При вводе пароля система будет вести себя так, будто ничего не происходит, но это не так. Ввод пароля осуществляется, но никак визуально не отображается.

Вводим пароль, нажимаем «Enter». Если пароль введён верно, то выведется приветственный текст и появится возможность выполнять команды на сервере:

```
PS C:\Users\super> ssh oleg@192.168.88.236 -p 422
oleg@192.168.88.236's password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.15.0-67-generic x86_64)
```

Перед развёртыванием БД необходимо проверить, установлены ли на сервере подходящие локали - это делается командой «*locale -a*».

```
root@underbed:/home# locale -a
C
C.utf8
en_US.utf8
POSIX
```

Убеждаемся, что в списке присутствует локаль «ru_RU.UTF-8».

Если такой локали нет, то выполняем команду *sudo locale-gen ru_RU.UTF-8*

```
root@underbed:/home# sudo locale-gen ru_RU.UTF-8
Generating locales (this might take a while)...
  ru_RU.UTF-8... done
Generation complete.
```

Добавляем новую локаль и обновляем данные о локалях командой *sudo update-locale.*

```
root@underbed:/home# sudo update-locale
root@underbed:/home#
```

Находим службу БД командой *systemctl | grep postgres*

```
root@underbed:/home# systemctl | grep postgres
  postgresql.service
          loaded active exited    PostgreSQL RDBMS
  postgresql@14-main.service
          loaded active running   PostgreSQL Cluster 14-main
  system-postgresql.slice
          loaded active active    Slice /system/postgresql
```

Нас интересует служба «*main*». Перезапускаем её командой *systemctl restart postresql@14-main.service*

После перезапуска службы переключаем пользователя на **postgres**. Если такого пользователя нет, то убеждаемся, что сервер БД развёрнут корректно (*Документация от DigitalOcean*).

Меняем пользователя на **postgres**.

Запускаем консоль управления **psql** и создаём новый экземпляр (инстанс) БД командой:

```
CREATE DATABASE finistreport_ifrs9 WITH
       TEMPLATE template0
       OWNER = postgres
       ENCODING = 'UTF8'
       LC_COLLATE = 'ru_RU.UTF-8'
       LC_CTYPE = 'ru_RU.UTF-8'
       TABLESPACE = pg_default
       CONNECTION LIMIT = -1;
```

Убеждаемся, что база создалась, командой ***\l***
Выходим из учётной записи **postgres** и переключаемся на **root**.

Далее необходимо развернуть базу, *отправленную в поставке*. Для этого выполняем команду:
`pg_restore -U postgres -d finistreport_ifrs9 dumpfilepath`

*! «**dumpfilepath**» - это путь к файлу БД*

Наша база данных развёрнута, необходимо создать пользователя, который будет *использован* для взаимодействия ПО с базой данных.

Переключаемся на пользователя **postgres** и его консоль **psql**.

Подключаемся к стандартной базе postgtresql – **template1**. Для этого выполняем команду *\connect template1* и создаём нужную роль командой:

```
create role gestuser with superuser nocreatedb nocreaterole noinherit login
noreplication nobypassrls password 'Cp12NsSg';
```

! В данном примере роль называется «gestuser», пароль (password) – меняем на свой.

```
postgres=# \c template1
You are now connected to database "template1" as user "postgres".
template1=# create role gestuser with superuser nocreatedb nocreaterole noinherit login noreplication nobypassrls password 'Cp12NsSg';
CREATE ROLE
```

Мы создали пользователя «gestuser», необходимо дать ему права доступа к базе данных приложения.

Отключаемся от текущей базы данный командой *exit* и подключаемся к базе данных приложения - в данном примере это «**finistreport_ifrs9**».

Последовательно выполняем следующие команды:

```
grant connect on database finistreport_ifrs9 to gestuser;
grant all privileges on schema dbo to gestuser;
grant all privileges on all tables in schema dbo to gestuser;
grant all privileges on all sequences in schema dbo to gestuser;
```

! При помощи данных команд мы дадим необходимые права пользователю «gestuser».

Для созданной базы данных указываем параметр *search_path* при помощи команды: `SET search_path TO dbo, public;`

! Данный параметр указывает БД, где по умолчанию необходимо производить поиск таблиц с данными, если в запросе не была указана конкретная схема БД.

Готово. Наша база данных настроена и готова к *использованию*.

6 Настройка приложений

6.1 FinistDigitalBank.Report.Worker

Настраивается при помощи двух файлов:

- **appsettings.json** - настройки для работы приложения:
- **FinistDigitalBank.Report.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.

6.1.1 Настройка файла appsettings.json

Ниже приведён пример содержимого файла appsettings.json.

```
{
    "AllowedHosts": "*",
    "Kestrel": {
        "EndpointDefaults": {
            "Protocols": "Http2"
        },
        "Endpoints": {
            "Http": {
                "Url": "http://0.0.0.0:5100"
            } //,
            //"Https": {
            //  "Url": "https://0.0.0.0:5101",
            //  // Uncomment certificate section to use specified certificate.
            //  "Certificate": {
            //          "Path": "GrpcCert1.pfx",
            //          "Password": "grpc1"
            //  }
            //}
        },
        "ServerOptions": { // GEst.Hosting options
            "Limits": {
                "MaxRequestBodySize": 157286400
            }
        }
    },
    "Logging": {
        "LogLevel": {
            "Default": "Information",
            "Microsoft": "Warning",
            "Microsoft.Hosting.Lifetime": "Information"
        },
        "EventLog": {
            "LogName": "Application", // GEst.Hosting options: by default Application
            "SourceName": "MyEvenLogSource", // GEst.Hosting options: by default application name (name of
entry assembly or startup assembly(for web))
            "LogLevel": {
                "Default": "Error",
                "Microsoft.Hosting.Lifetime": "Information",
                "GEst.Hosting.GEstStarterService": "Information",
                "GEst.Hosting.ProgramLogger": "Information"
            }
        }
    },
    "GEst.Hosting": {
        "WindowsService": { // only for Windows services
            "ServiceName": "MyWinService" // by default application name (name of entry assembly or startup
assembly(for web))
```

```
        },
        "CORS": { // CORS policies - will be added if specified at least one with nonempty name
            "Policies": [{
                    "Name": "GEst.WebClient.Api", // for api controllers in GEst.WebClient
                    "Origins": ["http://localhost:4205", "http://localhost"]
                }
            ]
        }
    }
}
```

! Данный пример можно использовать как начальную конфигурацию для сервера.

Краткое описание параметров файла:
AllowedHosts - адреса, с которых сервер принимает запросы.
Kestrel / EndpointDefaults - системная секция. Версия протокола gRPC.
Kestrel / Endpoints - содержит настройки прослушиваемых адресов. Детальная настройка прописана ниже в п. *HTTP/HTTPS WEBCLIENT – SERVER.*
ServerOptions / Limits / MaxRequestBodySize - максимальный размер тела запроса в байтах.
Logging - системная секция, отвечает за вывод различных логов.
GEst.Hosting / WindowsService - наименование службы при её регистрации в «Службах» операционной системы.
GEst.Hosting / CORS - политики CORS (Cross-origin resource sharing). Указываются источники, с которых приложение Server.Worker'а может принимать запросы.

HTTP/HTTPS WEBCLIENT – SERVER

В секции **Endpoints** находятся настройки выставленных адресов для gRPC. **Url** – адрес, по которому будет осуществляться общение с сервером.

! По умолчанию в комплект поставки входит тестовый сертификат – он находится в папке CommonFiles (FinistDigitalBank.Report.Server.Worker → CommonFiles)

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

6.1.2 FinistDigitalBank.Report.dll.config

6.1.2.1 Настройка подключения к базе данных

! Для настройки подключения ваша база данных должна быть подготовлена к использованию.

6.1.2.1.1 PostgreSQL

Ниже приведён пример настроек БД PostgreSQL:

```
...
<GEst.Server>
      <ConnectionString
value="Server=pgdev.esterdev.com;Port=5432;Database=database;User
ID=postgres;Password=Nt2CFWtGaUesu;Include Error Detail=true;SSL Mode=Require;Trust
Server Certificate=true" />
      <SqlDialect value="PostgreSql" />
</GEst.Server>
```

```
...
```

СТРОЧКА CONNECTIONSTRING

Параметры:
server - имя или IP-адрес сервера БД.
port – порт сервера БД.
database – наименование БД.
user id - пользователь БД, под которым будет осуществляться взаимодействие приложения с БД.
password - пароль пользователя.
SSL Mode - режим шифрования. Подробнее см. в *документации*. Обычно используется значение «Require» или «Allow».
trust server certificate – доверять сертификату сервера. Указывает, что серверный сертификат является доверенным.

! Применимо, когда на машине, где запущено сервер-приложение, сертификат не добавлен в доверенные - например, у банка есть самозаверенный сертификат.

СТРОЧКА SQLDIALECT

```
<SqlDialect value="PostgreSql" />
```

6.1.3 Настройка авторизации

! Данный способ авторизации доступен только на клиентских машинах Windows.

Для работы аутентификации через доменную учётную запись, необходимо добавить в секцию GEst.Server следующую строку.

```
<GEst.Server>
	...
	<WinAuthenticationServer port="5102"/>
	...
</GEst.Server>
```

Также пользователю системы в необходимо при настройке учетной записи добавить логин вида: «domain\user» и установить галку в поле «Windows-аккаунт».
В случае, если серверная ОС Linux – добавляем ***spn***:

```
<GEst.Server>
	...
	<WinAuthenticationServer port="5102"
spn="DEMOAPP/lxnet.esterdev.com@ESTERDEV.COM"/>
	...
</GEst.Server>
```

6.1.4 Настройка SPN

В данном блоке описывается порядок действий, совершенный на тестовой среде одним из разработчиков, для настройки spn. В случае, если регистрация spn производится по данной инструкции, необходим скрипт **kt.cmd**.

! Если файла нет, его необходимо запросить.

Прежде чем приступать к настройке, нужно убедиться, что:

DNS имя разрешается в обе стороны (имеются a- и ptr- записи в dns).

Контроллер домена - авторизованный источник времени в домене, либо все машины используют один источник времени.

Linux-машина добавлена в DNS, имя также разрешается в обе стороны.

Предоставлены следующие машины:

- Контроллер домена dc1.testlab.local Windows Server 2019, домен testlab.local (NETBIOS имя TESTLAB);
- Рабочая станция Windows w10.testlab.local Windows 10 21H2;
- Рабочая станция Linux u20.testlab.local Ubuntu 2004.

На контроллере домена с правами администратора домена выполняем следующие действия:

Создаём сервисного пользователя labsvb, указываем пароль и опции **password never expires** и **user can't change password**. В настройках пользователя указываем **This account supports Kerberos AES 128 bit encryption** и **256 bit encryption**.

Скрипт для присваивания SPN и формирования keytab-файла называется **kt.cmd**, должен быть приложен к письму.

В случае, если файла нет, его необходимо запросить.

В начале файла меняем строчки:

```
set "KUSER=TESTLAB\labsvc"
set "KPASS=pass123"
set "KRESETPASS=no"
set "KSPN=DEMOAPP/u20.testlab.local;DEMOAPP/u20"
```

KUSER – имя сервисного пользователя.
KPASS – пароль сервисного пользователя.
KSPN – список spn для регистрации.
В данном примере мы зарегистрируем два SPN:

- *DEMOAPP/u20.testlab.local@TESTLAB.LOCAL*
- *DEMOAPP/u20@TESTLAB.LOCAL*

Создаём каталог на контроллере домена, например «keytabs», подкладываем туда изменённый скрипт **kt.cmd** и выполняем его.
В результате должны быть зарегистрированы spn-ы, а в каталоге сформироваться файл labsvc.keytab.
Файл .keytab подкладываем на linux-машину.
Проверить, что SPN зарегистрирован, можно командой `setspn -L labsvc`.
На Linux-машине настраиваем FQDN, имя хоста - u20.testlab.local, в качестве Dns-сервера указываем контроллер домена.
Добавляем SPN в секцию **WinAuthenticationServer:**

```
<GEst.Server>
      ...
      <WinAuthenticationServer port="5102"
spn="DEMOAPP/u20.testlab.local@TESTLAB.LOCAL"/>
```

```
        ...
</GEst.Server>
```

6.2 Настройка LDAP сервера

В серверном конфигурационном файле (FinistDigitalBank.Report.dll.config) есть возможность настроить LDAP сервер для аутентификации пользователей через доменные учётные данные. LDAP серверов может быть несколько.

```
<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'у
System.DirectoryServices.Protocols.AuthType; по умолчаннию 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">
        <GEstAuthentication>
            <!-- 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))" />
        </GEstAuthentication>
    </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)">
        <GEstAuthentication>
            <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
.113556.1.4.803:=2)))" />
        </GEstAuthentication>
    </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">
        <GEstAuthentication>
            <Bind dynamic="true" dn="uid=%%login%%,ou=Users,dc=ldaptest,dc=com" />
            <Search root="dc=ldaptest,dc=com" filter="(&amp;(uid=%%login%%)(objectClass=posixAccount))" />
        </GEstAuthentication>
```

```
    </Server>
</Ldap>
```

Для возможности входа без указания доменного имени, необходимо указать атрибут **defaultServer** - уникальный ключ сервера, значение из секции **Server/@key**. Этот домен будет использоваться как основной. Пользователи, **расположенные в нём**, могут не указывать доменное имя при авторизации.

```
<GEst.Server>
	...
	<Ldap defaultServer="orgdomain" ...>
		<Server key="orgdomain">
		...
		</Server>
		<Server key="otherdomain" ...>
		...
		</Server>
	</Ldap>
	...
</GEst.Server>
```

6.3 FinistDigitalBank.Report.UserArm

6.3.1 FinistDigitalBank.Report.UserArm.config

Меняется только одна настройка, в которой указываем адрес и порт сервера:

```
...
<GEst.Common>
	<HostAddress value="http://192.168.0.51:5700" />
</GEst.Common>
...
```

Адрес сервера указывается, как *http://адрес:порт*. Пример: «**http://192.168.0.51:5700**»
Порт настраивается в *серверной конфигурации*.

7 Регистрация служб

7.1 Linux

7.1.1 Регистрация приложений

В ОС Linux службы (демоны) регистрируются путём создания файла-юнита. Все операции должны быть выполнены root-пользователем (команда sudo). Данный пользователь также должен иметь доступ на редактирование каталога (права на запись и чтение), в котором располагается ПО. Пример юнита для FinistDigitalBank.Report.Worker60.dll

```
[Unit]
Description= FinistDigitalBank.Report Service

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

# systemd will run this executable to start the service
ExecStart=/usr/bin/dotnet PATH_TO_WORKER_DLL &

# to query logs using journalctl, set a logical name here
SyslogIdentifier= FinistDigitalBank.Report.Worker

# 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/
FinistDigitalBank.Report.Server.Worker' to take ownership of the folder and files,
#       Use 'chmod +x /opt/EM/ FinistDigitalBank.Report.Server.Worker/
FinistDigitalBank.Report.Server.Worker.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 службы.

ExecStart – исполняемая команда при запуске службы через systemctl. Указываем путь до исполняемой dll.
Конфигурационный файл применим как к службе сервера, так и к службе веб-клиента, необходимо лишь заменить пути до запускаемых файлов.
User – пользователь, под которым будет осуществлён запуск и исполнение приложения.

! Поля, отмеченные красным, необходимо заполнить в соответствии с вашей конфигурацией.

Для создания файла выполняем консольную команду:
`touch /etc/systemd/system/FinistDigitalBank.Report.Server.Server.Worker.service`
Данная команда создает пустой файл FinistDigitalBank.Report.Worker.service. Далее, при помощи команды:
`nano /etc/system/system/ FinistDigitalBank.Report.Server.Worker.service`
открываем данный файл в редакторе nano и вставляем (ПКМ) конфигурацию юнита из примера выше, не забыв заменить пути до приложения **PATH_TO_APP_DIRECTORY** и **PATH_TO_WORKER_DLL.**
После проделанных действий вызываем команду:
`systemctl daemon-reload`
которая обновляет и подтягивает конфигурации system.
Далее, мы можем произвести запуск службы командой:
`systemctl start FinistDigitalBank.Report.Server.Worker.service`
Для проверки статуса службы необходимо использовать команду:
`systemctl status FinistDigitalBank.Report.Server.Worker.service`
В случае, если запуск службы произведён успешно, будет выведено сообщение

```
adm@lxnet:~$ systemctl status EventManager.Server.Worker
● EventManager.Server.Worker.service - Event manager server service
     Loaded: loaded (/etc/systemd/system/EventManager.Server.Worker.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2023-04-26 22:06:35 MSK; 10s ago
   Main PID: 87610 ((dotnet))
     CGroup: /system.slice/EventManager.Server.Worker.service
             └─87610 /usr/bin/dotnet /home/adm/lxnet/backend/server/EventManager.Server.Worker.dll &
adm@lxnet:~$ systemctl status EventManager.Server.Worker
● EventManager.Server.Worker.service - Event manager server service
     Loaded: loaded (/etc/systemd/system/EventManager.Server.Worker.service; enabled; vendor preset: enabled)
     Active: failed (Result: exit-code) since Wed 2023-04-19 10:21:07 MSK; 1 weeks 0 days ago
    Process: 7265 ExecStart=/usr/bin/dotnet /home/adm/lxnet/backend/server/EventManager.Server.Worker.dll & (code=exite>
   Main PID: 7265 (code=exited, status=56)
lines 1-5/5 (END)
```

Важно проверить статус службы спустя несколько минут, так как для запуска приложения требуется время.

8 Настройка общего доступа до каталога с Linux на Windows

Перед конфигурацией linux-системы создадим нового пользователя windows с паролем и дадим ему право на чтение каталога. Это нужно для последующего использования данной учётной записи в linux.

Для того чтобы пользоваться общим windows-каталогом на linux сервере, необходимо установить утилиту **cifs-utils.** Она позволит подключить (mount) общий каталог как отдельный диск. Под супер пользователем исполняем команду: apt install cifs-utils.

```
root@underbed:/home# apt install cifs-utils
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  smbclient winbind
The following NEW packages will be installed:
  cifs-utils
0 upgraded, 1 newly installed, 0 to remove and 55 not upgraded.
Need to get 95.7 kB of archives.
After this operation, 335 kB of additional disk space will be used.
Get:1 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 cifs-utils amd64 2:6.14-1ubuntu0.1 [95.7 kB]
Fetched 95.7 kB in 0s (818 kB/s)
Selecting previously unselected package cifs-utils.
(Reading database ... 172379 files and directories currently installed.)
Preparing to unpack .../cifs-utils_2%3a6.14-1ubuntu0.1_amd64.deb ...
Unpacking cifs-utils (2:6.14-1ubuntu0.1) ...
Setting up cifs-utils (2:6.14-1ubuntu0.1) ...
update-alternatives: using /usr/lib/x86_64-linux-gnu/cifs-utils/idmapwb.so to provide /etc/cifs-utils/idmap-plugin (idmap-plugin) in auto mode
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning candidates...
Scanning processor microcode...
Scanning linux images...
```

Далее создадим каталог, который будет являться точкой подключения, командой mkdir /mnt/win-share.

8.1 Тестовое подключение каталога

После создания каталога, необходимо «примонтировать» общий каталог, это делается командой:

mount -t cifs -o username=%win пользователь% //%ip windows машины%/%каталог% /mnt/win_share

```
root@underbed:/home# mount -t cifs -o username=oleg //192.168.88.230/shared /mnt/win-share
Password for oleg@//192.168.88.230/shared:
root@underbed:/home#
```

Готово. Проверим, что всё получилось, командой **ls**.

```
root@underbed:/home# ls /mnt/win-share/
favicon.ico
root@underbed:/home#
```

8.2 Конфигурация авто-монтирования

Для того, чтобы после перезапуска системы каталог оставался доступен, необходимо указать системе, что его нужно подключать в момент запуска системы. Делается это через изменение конфигурации файловой системы, файл /etc/fstab.

Прежде чем приступать к изменению fstab создадим файл, который будет содержать данные об авторизации в windows. Создадим каталог и файл с данными командами:

mkdir /etc/win-creds/
touch /etc/win-creds/share

```
root@underbed:/home# mkdir /etc/win-creds/
root@underbed:/home# touch /etc/win-creds/share
root@underbed:/home# nano /etc/win-creds/share
```

Открываем его при помощи **nano** (или любого другого редактора):

```
 GNU nano 6.2                    /etc/win-creds/share *
username=oleg
password=oleg-password
domain=domain
```

Указываем логин/пароль/домен пользователя (если существует).

После изменения файла настроем доступы до каталога. Запретим всем пользователям, кроме root, как-либо взаимодействовать с файлами win-creds. Выполняем команды:

chown root: /etc/win-creds

chmod 600 /etc/win-creds

```
root@underbed:/home# chown root: /etc/win-creds
root@underbed:/home# chmod 600 /etc/win-creds
root@underbed:/home# ls -l /etc/win-creds
total 4
-rw-r--r-- 1 root root 51 Mar 24 11:28 share
```

! После исполнения проверяем, что правила применились, при помощи ls -l /etc/win-creds

Теперь перейдём к редактированию fstab файла. Добавляем строчку:

//%ip win-машины%/%общедоступный каталог% /mnt/win-share cifs credentials=/etc/win-creds/share,file_mode=0755,dir_mode=0755 0 0

```
//192.168.88.230/shared /mnt/win-share cifs credentials=/etc/win-creds/share,file_mode=0755,dir_mode=0755 0 0
```

Данная запись будет использована при запуске системы, монтируем вручную командой mount /mnt/win-share.

```
root@underbed:/home# mount /mnt/win-share
root@underbed:/home# ls /mnt/win-share
favicon.ico
root@underbed:/home#
```

Готово. Теперь данный каталог будет монтироваться при каждом запуске системы. Также, для удобства использования в веб-сервере, можно создать символическую ссылку, чтобы не ссылаться напрямую в /mnt/.