

**Инструкция по установке
программного обеспечения «РОМОНА
ВИЗИОНЕР» по обнаружению объектов на
снимках беспилотного летательного аппарата
(БЛА)**

Южно-Сахалинск

2025

Данная инструкция предназначена для Linux администраторов и описывает установку системы на Ubuntu Linux 22.04

1. Программный состав системы

- Веб-приложение – пользовательский интерфейс системы
- ML-backend – сервис для обработки изображений
- Keycloak – сервис аутентификации и авторизации
- Rule-engine – сервис для поиска нарушений

1.2 Установка веб-приложения

1.2.1 Подготовка окружения

Для запуска компонента необходимы: Python 3.9+, PostgreSQL 16+ и доступ в интернет для скачивания зависимостей.

Установка и настройка PostgreSQL 16

```
sudo apt update
```

Добавить репозиторий PostgreSQL:

```
sudo sh -c 'echo "deb https://apt.PostgreSQL.org/pub/repos/apt $(lsb_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'
```

Импортировать ключ репозитория: `wget --quiet -O -`

```
https://www.PostgreSQL.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
```

Обновить локальный индекс: `sudo apt update`

Установить PostgreSQL 16: `sudo apt install PostgreSQL-16 -y`

Управление сервисом postgresql

```
sudo systemctl start postgresql
```

```
sudo systemctl enable postgresql
```

Далее необходимо создать пользователя, БД и выдать права пользователю на БД.

```
Sudo -i -u postgres psql
```

```
CREATE USER romona WITH ENCRYPTED PASSWORD '12345';
```

```
CREATE DATABASE romona;
```

```
GRANT ALL PRIVILEGES ON DATABASE romona TO romona;
```

Установка Python

В Ubuntu 22.04 уже установлена подходящая версия python

Версию можно проверить `python3 --version`

1.2.2 Установка и настройка компонента

Создать директорию командой: `mkdir -R /srv/romona`

Распаковать архив в созданную директорию: `unzip romona.zip -d /srv/romona`

Установить компонент виртуального окружения Пи: `sudo apt install python3.12-venv`

В распакованом архиве в папке `/srv/romona/romona` создать виртуальное окружение командой `python3 -m venv .venv`

Далее активировать виртуальное окружение командой: `source .venv/bin/activate`

Установить pip: `sudo apt install python3-pip`

Обновить pip: `pip install --upgrade pip`

Установить зависимости:

```
pip install psycopg2-binary
sudo apt install libpq-dev gcc
```

```
pip install -e .
```

BUG THIS HERE!

Запустить файл `./migrate_collectstatic.sh` и далее соглашаться на все

Деактивировать виртуальное окружение командой `deactivate`

В корне папки `romona` находится файл `romona.service`, его надо поместить в `/etc/systemd/system`

Дальнейшее управление сервисом осуществляется стандартными командами `systemctl start/stop/restart/status/и т.п. romona`.

Также рекомендуется выполнить команду `systemctl enable romona` для автоматического запуска сервиса при загрузке сервера.

Описание параметров в дескрипторе сервиса (файл `/etc/systemd/system/romona.service`):

`Environment=POSTGRE_USER=romona` - пользователь СУБД

`Environment=POSTGRE_PASSWORD=12345` – пароль пользователя СУБД

`Environment=KEYCLOAK_CLIENT_SECRET=wrSubQ1BJNXytmCSfR7j3oOS5` – берется из Keycloak

`Environment=KEYCLOAK_CLIENT_ID=romona_visioner_app` – id клиента из Keycloak

`Environment=KEYCLOAK_AUTH_REDIRECT_URL=<адрес по которому будет доступно веб-приложение>/auth_redirect` – URL куда будет осуществляться редирект после успешной аутентификации

`ExecStart=/bin/bash -c '/srv/romona/romona/.venv/bin/python label_studio/manage.py runserver 0.0.0.0:8000 --noreload'` – команда запуска приложения на 8000 порте

`WorkingDirectory=/srv/romona/romona` – рабочая директория приложения

1.2.3 Обновление компонента

Остановить приложение командой: `systemctl stop romona`

Удалить содержимое директории `/srv/romona/romona`, кроме `.venv`:

```
cd /srv/romona/romona
```

```
find . -not -name '.venv' -delete
```

Распаковать архив в директорию: `unzip romona.zip -d /srv/romona`

Далее активировать виртуальное окружение командой: `source .venv/bin/activate`

Обновить зависимости: `pip install -e .`

Запустить файл `./migrate_collectstatic.sh` и далее соглашаться на все

Деактивировать виртуальное окружение командой `deactivate`

Сверить содержимое файла `romona.service` со старой версией в `/etc/systemd/system/romona.service` и при появлении новых параметров добавить их в старую версию.

Запустить приложение командой: `systemctl start romona`

1.3 Установка ML-backend

1.3.1 Подготовка окружения

Для запуска компонента необходимы: Python 3.9+, Nvidia driver $\geq 525.60.13$ и доступ в интернет для скачивания зависимостей.

Установка Python

В Ubuntu 22.04 уже установлена подходящая версия python

Версию можно проверить `python3 --version`

Установка драйверов Nvidia

Версию для установки, удовлетворяющую требованиям, можно найти в дефолтных репозиториях Ubuntu и установить командой: `sudo apt install nvidia-driver-525`

Проверить корректность установки можно командой: `nvidia-smi`

1.3.2 Установка и настройка компонента

Распаковать архив в созданную директорию: `unzip romona-ml-backend.zip -d /srv/romona`

В распакованном архиве в папке `/srv/romona/romona-ml-backend` создать виртуальное окружение командой `python3 -m venv .venv`

Далее активировать виртуальное окружение командой: `source .venv/bin/activate`

Обновить pip: `pip install --upgrade pip`

Установить зависимости: `pip install -r requirements.txt`

Деактивировать виртуальное окружение командой `deactivate`

Модель распаковать из архива: `unzip models.zip -d /srv/romona.`

Обновить в `MODEL_PATH` и `MODEL_VERSION` путь до актуальной модели.

В корне папки `romona` находится файл `romona-ml-backend.service`, его надо поместить в `/etc/systemd/system`

Дальнейшее управление сервисом осуществляется стандартными командами `systemctl start/stop/restart/status/и т.п. romona-ml-backend.`

Также рекомендуется выполнить команду `systemctl enable romona-ml-backend` для автоматического запуска сервиса при загрузке сервера.

Описание параметров в дескрипторе сервиса (файл `/etc/systemd/system/romona-ml-backend.service`):

`Environment=MODEL_PATH=/srv/romona/models/checkpoint-122000` – путь до модели, которая будет использоваться, находится в `romona/models`

`Environment=MODEL_VERSION=0.0.5` – версия модели, необходимо изменять при обновлениях модели, для маркировки предсказаний

`Environment=APP_HOST=http://192.168.10.38:8000` - адрес, где развернуто веб-приложение(визионер)

`Environment=PYTHONPATH=/srv/romona/romona-ml-backend` – путь до python файлов.

`ExecStart=/bin/bash -c '/srv/romona/romona-ml-backend/.venv/bin/python _wsgi.py'` – команда запуска `romona-ml-backend` сервиса

`WorkingDirectory=/srv/romona/romona-ml-backend/seg_ml_backend` – рабочая директория приложения

1.3.3 Обновление компонента

Остановить приложение командой: `systemctl stop romona-ml-backend`

Удалить содержимое директории `/srv/romona/romona-ml-backend`, кроме `.venv`:

```
cd /srv/romona/romona-ml-backend
```

```
find . -not -name '.venv' -delete
```

Распаковать архив в директорию: `unzip romona-ml-backend.zip -d /srv/romona`

Далее активировать виртуальное окружение командой: `source .venv/bin/activate`

Обновить зависимости: `pip install -r requirements.txt`

Деактивировать виртуальное окружение командой `deactivate`

Модель распаковать из архива: `unzip models.zip -d /srv/romona`.

Обновить в `MODEL_PATH` и `MODEL_VERSION` путь до актуальной модели.

Свернуть содержимое файла `romona-ml-backend.service` со старой версией в `/etc/systemd/system/romona-ml-backend.service` и при появлении новых параметров добавить их в старую версию.

Запустить приложение командой: `systemctl start romona-ml-backend`

1.4 Установка Keycloak

1.4.1 Подготовка окружения

Для запуска Keycloak требуется Java 21+, PostgreSQL 16+

Установка PostgreSQL

Установка PostgreSQL описана выше, единственное отличие, необходимо создать БД и пользователя с именем keycloak:

```
Sudo -i -u postgres postgres
CREATE USER keycloak WITH ENCRYPTED PASSWORD '12345';
CREATE DATABASE keycloak;
GRANT ALL PRIVILEGES ON DATABASE keycloak TO keycloak;
```

Установка OpenJDK

Установить командой: `sudo apt install openjdk-21-jdk`

Проверить версию java: `java -version`

1.4.2 Установка и настройка компонента

Создать директорию командой: `mkdir -R /srv/admin`

Распаковать архив в созданную директорию: `unzip keycloak.zip -d /srv/admin`

Далее необходимо задать логин/пароль администратора. В файле `/srv/admin/keycloak-26.0.0/start.sh` задать логин/пароль администратора в строках, предварительно раскомментировав их:

```
export KEYCLOAK_ADMIN=<username>
```

```
export KEYCLOAK_ADMIN_PASSWORD=<password>
```

Также в этом файле необходимо задать `hostname` и если потребуется, то `http-port` и параметры соединения с БД(с префиксом `db-`)

В корне папки `romona` находится файл `keycloak.service`, его надо поместить в `/etc/systemd/system`

Дальнейшее управление сервисом осуществляется стандартными командами `systemctl start/stop/restart/status/и т.п. keycloak`.

Также рекомендуется выполнить команду `systemctl enable keycloak` для автоматического запуска сервиса при загрузке сервера.

1.5 Rule-engine

1.5.1 Подготовка окружения

Для запуска компонента необходимы: Python 3.9+ и доступ в интернет

Установка Python была приведена выше.

1.5.2 Установка и настройка компонента

Распаковать архив в созданную директорию: `unzip rule-engine.zip -d /srv/romona`

В распакованном архиве в папке `/srv/romona/rule-engine` создать виртуальное окружение командой: `cd /srv/romona/rule-engine`

```
python3 -m venv .venv
```

Далее активировать виртуальное окружение командой: `source .venv/bin/activate`

Обновить pip: `pip install --upgrade pip`

Установить Poetry: `pip install poetry`

Установить зависимости: `poetry install`

Деактивировать виртуальное окружение командой `deactivate`

В корне папки `romona` находится файл `rule-engine.service`, его надо поместить в `/etc/systemd/system`

Дальнейшее управление сервисом осуществляется стандартными командами `systemctl start/stop/restart/status` и т.п. `rule-engine`.

Также рекомендуется выполнить команду `systemctl enable rule-engine` для автоматического запуска сервиса при загрузке сервера.

Описание параметров в дескрипторе сервиса (файл `/etc/systemd/system/rule-engine.service`):

`Environment=AUTH_URL=<адрес, по которому доступна Keycloak>/realms/romona/protocol/openid-connect/token` – URL по которому будет доступен токен в Keycloak

`Environment=LABEL_STUDIO_URL=https://romona.nemosoft.ru` – URL по которому будет доступно веб-приложение

`Environment=KEYCLOAK_SERVER_URL=<адрес, по которому доступна Keycloak>/auth` – URL аутентификации в Keycloak

`Environment=KEYCLOAK_CLIENT_ID=romona_visioner_app` – идентификатор клиента в Keycloak

`Environment=KEYCLOAK_REALM=romona` – realm в Keycloak

`Environment=KEYCLOAK_CLIENT_SECRET=wrSubPl1mgYQ5` – секретный ключ клиента Keycloak

`Environment=LABEL_STUDIO_LOGIN=rule-engine` – служебная УЗ в Keycloak

`Environment=LABEL_STUDIO_PASSWORD=12345` – пароль для служебной УЗ в Keycloak

`Environment=NPB_SERVER_PORT=8088` – порт на котором будет доступно отладочное веб-приложение

`Environment=NBP_DEFAULT_SCHEDULE_INTERVAL_MINUTES=1` – интервал, через который будут запускаться задания на поиск нарушений

`Environment=NBP_EXEC_JOBS_BEFORE_SCHEDULE=True` – стартовать поиск нарушений сразу после запуска сервиса

`Environment=PX_IMAGE_WIDTH_THRESHOLD=5000` – параметр используется для вычисления расстояний

`Environment=PX_DEFAULT_IMAGE_WIDTH=7952` – параметр используется для вычисления расстояний

`Environment=PX_DEFAULT_CAMERA_SENSOR_WIDTH=36` – ширина матрицы камеры

`Environment=TASKS_PER_CYCLE_LIMIT=10000000` – максимальное количество задач, обрабатывающихся за цикл

`ExecStart=/bin/bash -c '/srv/romona/rule-engine/.venv/bin/python run.py'` – команда запуска `rule engine`

WorkingDirectory=/srv/romona/rule-engine – рабочая директория сервиса

Примечание: также для данного компонента необходимо настроить техническую учетную запись. (как это сделать, описано в документе «Быстрая настройка Keycloak»). После настройки УЗ необходимо залогиниться в веб-приложение(визионер) и убедиться, что все работает и доступно под этой УЗ.

1.5.3 Обновление компонента

Остановить приложение командой: `systemctl stop rule-engine`

Удалить содержимое директории `/srv/romona/rule-engine`, кроме `.venv`:

```
cd /srv/romona/rule-engine
```

```
find . -not -name '.venv' -delete
```

Распаковать архив в директорию: `unzip rule-engine.zip -d /srv/romona`

Далее активировать виртуальное окружение командой: `source .venv/bin/activate`

Обновить зависимости: `poetry install`

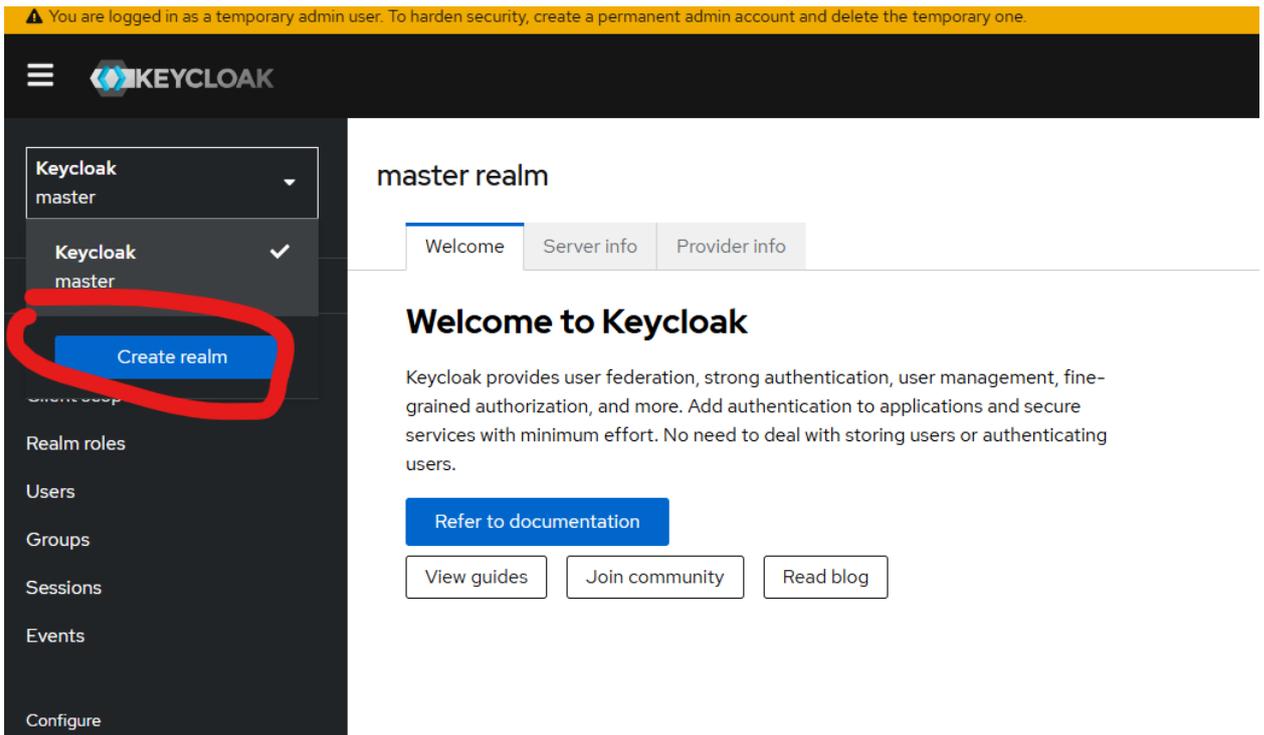
Деактивировать виртуальное окружение командой `deactivate`

Сверить содержимое файла `rule-engine.service` со старой версией в `/etc/systemd/system/rule-engine.service` и при появлении новых параметров добавить их в старую версию.

Запустить приложение командой: `systemctl start rule-engine`

Быстрая настройка Keycloak

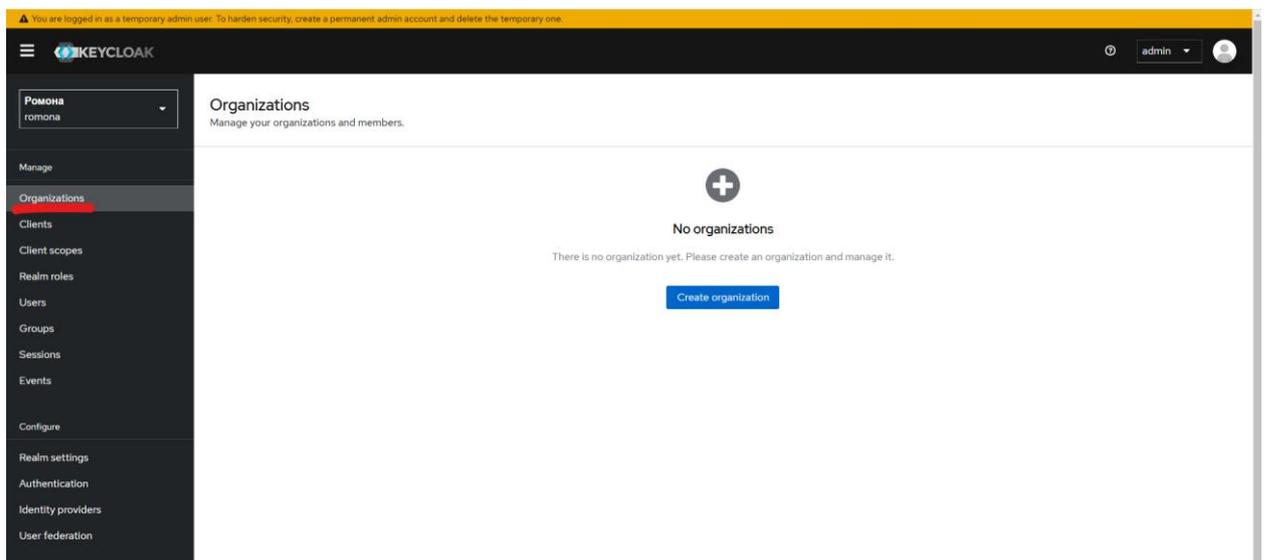
Создать realm `romona`



Далее в диалоге создания необходимо указать файл с основными настройками realm (realm-export.json) и нажать кнопку Create

После этого будет создан realm с именем ромона

Далее необходимо создать Организацию



Organizations > Create organization

Create organization

Name *

Alias

Domain [Add domain](#)

Redirect URL

Description

[Save](#) [Cancel](#)

В последующем при создании пользователя важно привязывать его к организации, чтобы система показывала только принадлежащие организации пользователя проекты и т.п.

Далее необходимо создать пользователя

You are logged in as a temporary admin user. To harden security, create a permanent admin account and delete the temporary one.

KEYCLOAK admin

Users

Users are the users in the current realm. [Learn more](#)

User list Permissions

Default search Search user [Add user](#) Delete user Refresh

<input type="checkbox"/>	Username	Email	Last name	First name	
<input type="checkbox"/>	service-account-realm-management	● -	-	-	⋮
<input type="checkbox"/>	service-account-romona_visioner_app	● -	-	-	⋮
<input type="checkbox"/>	service-account-secret-romona-visioner-app	● -	-	-	⋮
<input type="checkbox"/>	service-account-test	● -	-	-	⋮

1-4 < >

The screenshot shows the Keycloak user creation interface. On the left is a dark sidebar with a menu containing: Ромона, Manage, Organizations, Clients, Client scopes, Realm roles, Users (highlighted), Groups, Sessions, Events, Configure, Realm settings, Authentication, Identity providers, and User federation. The main content area has a top bar with 'KEYCLOAK' and a hamburger menu. Below this is a dropdown for 'Ромона' and a 'Required user actions' dropdown. A toggle for 'Email verified' is set to 'Off'. The 'General' section contains input fields for 'Username' (filled with 'superuser'), 'Email', 'First name', and 'Last name'. The 'User metadata' section includes fields for 'Отчество' (filled with 'patronymic') and 'Номер телефона' (filled with 'phone'), along with a 'Join Groups' button. A 'Jump to section' sidebar on the right lists 'General' and 'User metadata'. At the bottom are 'Create' and 'Cancel' buttons.

Для создания пользователя достаточно задать имя пользователя и пароль. Можно так же заполнить и другие поля.

The screenshot shows the 'User details' page for a user named 'superuser'. The top right corner shows the user 'admin' and a profile icon. The page has a breadcrumb 'Users > User details' and a status 'Enabled' with an 'Action' dropdown. Below the breadcrumb are tabs for 'Details', 'Credentials', 'Role mapping', 'Groups', 'Consents', 'Identity provider links', and 'Sessions'. The 'Credentials' tab is active, showing a large plus sign icon and the text 'No credentials'. Below this, it says 'This user does not have any credentials. You can set password for this user.' and a 'Set password' button.

Set password for superuser

Password *

Password confirmation *

Temporary Off

На следующей вкладке пользователю необходимо назначить хотя бы 1 роль.

Users > User details
 superuser Enabled Action

Details Credentials **Role mapping** Groups Consents Identity provider links Sessions

Hide inherited roles Assign role Unassign Refresh

<input type="checkbox"/>	Name	Inherited	Description
<input type="checkbox"/>	default-roles-romona	False	role_default-roles

Данная инструкция предполагает, что настройка ведется для клиента с именем romona_visioner_app, поэтому новому пользователю назначаем роль от этого клиента

Assign roles to superuser

<input type="checkbox"/>	realm-management	view-identity-providers	role_view-identity-providers
<input type="checkbox"/>	realm-management	view-realm	role_view-realm
<input type="checkbox"/>	realm-management	view-users	role_view-users
<input type="checkbox"/>	romona_visioner_app	uma_protection	–
<input type="checkbox"/>	romona_visioner_app	Администратор ИБ	–
<input type="checkbox"/>	romona_visioner_app	Оператор БЛА	Оператор, который осуществляет управление БВС
<input type="checkbox"/>	romona_visioner_app	Разметчик	1231234444
<input type="checkbox"/>	romona_visioner_app	Руководитель проекта	–
<input checked="" type="checkbox"/>	romona_visioner_app	Суперпользователь	У этого пользователя нет ограничений
<input type="checkbox"/>	romona_visioner_app	тест всемогущий	Тестовая роль
<input type="checkbox"/>	romona_visioner_app	тест полный (без мандатов)	–
<input type="checkbox"/>	romona_visioner_app	Топ-менеджер	–
<input type="checkbox"/>	secret-romona-visioner-app	uma_protection	–
<input type="checkbox"/>	secret-romona-visioner-app	Администратор ИБ	–
<input type="checkbox"/>	secret-romona-visioner-app	Оператор БЛА	–
<input type="checkbox"/>	secret-romona-visioner-app	Разметчик	–
<input type="checkbox"/>	secret-romona-visioner-app	Руководитель проекта	–
<input type="checkbox"/>	secret-romona-visioner-app	Суперпользователь	–

Assign

Cancel

Далее в клиенте `romona_visioner_app` необходимо проставить актуальные для конкретного деплоя Root URL, Web origins, Admin URL

The screenshot shows the Keycloak Admin Console interface for configuring a client named "Romona Визьюнер". The left sidebar contains navigation options like "Manage", "Organizations", "Clients", "Client scopes", "Realm roles", "Users", "Groups", "Sessions", "Events", "Configure", "Realm settings", "Authentication", "Identity providers", and "User federation". The main content area is titled "Access settings" and includes the following fields:

- Name:** Ромона Визьюнер
- Description:** (empty)
- Always display in UI:** On
- Root URL:** https://romona.nemosoft.ru
- Home URL:** /
- Valid redirect URIs:** /auth_redirect
- Valid post logout redirect URIs:** +
- Web origins:** https://romona.nemosoft.ru
- Admin URL:** https://romona.nemosoft.ru

At the bottom of the configuration form, there are "Save" and "Revert" buttons. On the right side, there is a "Jump to section" menu with options: General settings, Access settings, Capability config, Login settings, and Logout settings.

Для настройки некоторых компонентов необходим client secret, его можно взять здесь:

KEYCLOAK

romona

romona_visioner_app OpenID Connect Enabled Action

Clients > Client details

Settings Keys **Credentials** Roles Client scopes Authorization Service accounts roles Sessions Permissions Advanced

Client Authenticator Client Id and Secret

Save

Client Secret Regenerate

Registration access token Regenerate