**Информационная система «Face2»\***

Руководство по развертыванию

Версия 1.0

АО «Социальная Карта»

2025 г.

Оглавление

[Термины и определения 4](#_Toc201328772)

[Требования к окружению 5](#_Toc201328773)

[Требования к поставщику услуг 5](#_Toc201328774)

[Интеграция с ЕСИА 5](#_Toc201328775)

[Интеграция с ЕБС 5](#_Toc201328776)

[Зависимость от СМЭВ 5](#_Toc201328777)

[Требование к хранению ЦП 6](#_Toc201328778)

[Системные требования 7](#_Toc201328779)

[Системные требования 8](#_Toc201328780)

[Системные требования для серверов БД 8](#_Toc201328781)

[Системные требования для серверов приложений и модулей Системы 8](#_Toc201328782)

[Системные требования для запуска инсталлятора 8](#_Toc201328783)

[Служебные учетные записи 8](#_Toc201328784)

[Варианты развертывания ПО Face2 10](#_Toc201328785)

[Варианты развертывания 10](#_Toc201328786)

[Роли серверов 10](#_Toc201328787)

[Для установки на выделенные серверы: 10](#_Toc201328788)

[Развертывание для демонстрации возможностей Системы 11](#_Toc201328789)

[Подготовка образа: 11](#_Toc201328790)

[Важные примечания: 11](#_Toc201328791)

[Развертывание ПО Face2 в Kubernetes 12](#_Toc201328792)

[Подготовка среды для развертывания 12](#_Toc201328793)

[Подготовка дистрибутива 12](#_Toc201328794)

[Загрузка docker-образов в хранилище 13](#_Toc201328795)

[Загрузка docker-образов в хранилище сервера с ролью Установщик 13](#_Toc201328796)

[Установка утилит kubectl и helm 14](#_Toc201328797)

[Установка утилит kubectl и helm на сервере с ролью Установщик 14](#_Toc201328798)

[Конфигурация параметров стенда перед установкой 14](#_Toc201328799)

[Проверка работоспособности ПО Face2 16](#_Toc201328800)

[Просмотр статуса здоровья для каждого сервиса и модуля 16](#_Toc201328801)

[Настроенные проверки здоровья 16](#_Toc201328802)

[Интерпретация результатов 17](#_Toc201328803)

# [Термины и определения](https://kb.akbars.ru/pages/viewpage.action?pageId=345182135)

Термины, сокращения и определения, используемые в инструкции для развертывания ПО Face2:

* **Helm**: средство установки *kubernetes*-приложений; совокупность шаблонов *kubernetes*и подстанавливаемых значений, используемые для установки Системы.
* **Инсталлятор Системы**, также **Инсталлятор**: совокупность *ansible*-роли, набора переменных, используемых для развёртывания Системы на стенде, и *ansible-playbook* — yaml-файла, в котором указано, какие задачи и на каких серверах будут выполняться.
* **Inventory**: директория, используемая Инсталлятором, содержащая поддиректории с именами, совпадающими с именами стендов. В каждой из поддиректорий находятся файлы, содержащие параметры, зависящие от адреса сервера — переменные *ansible*и их значения.
* **Имя стенда**: обозначение стенда Системы, используемое в Inventory и для идентификации стенда. Обычно, но необязательно, совпадает с полным доменным именем сервера с ролью Сервер Приложений.
* **Неймспейс Kubernetes**, также **Неймспейс**или **Пространство имён**: виртуальный кластер внутри физического кластера Kubernetes, синоним термина стенд.
* **Стенд**: абстракция, содержащая в себе совокупность всех серверов, обеспечивающих работу одного экземпляра продукта Система.

# [Требования к окружению](https://kb.akbars.ru/pages/viewpage.action?pageId=345181932)

Система Face2 для получения информации о пользователе интегрируется со следующими государственными системами:

* Федеральная государственная информационная система «Единая система идентификации и аутентификации», далее ЕСИА;
* Государственная информационная система «Единая система идентификации и аутентификации физических лиц с использованием биометрических персональных данных», далее ЕБС.

## Требования к поставщику услуг

Система должна быть развернута на серверах поставщика услуг и соответствовать требования предъявляемыми:

* Федеральный закон от 29.12.2022 № 572-ФЗ
* Указание Банка России от 25 сентября 2023 г. N 6541-У
* Методические рекомендации Банка России от 20 декабря 2023 г. № 18-МР
* Методические рекомендации Банка России от 23 декабря 2023 г. № 19-МР
* Поставщик услуг должен быть зарегистрирован в ЕСИА как Юридическое лицо
* Поставщик услуг должен быть аккредитован для возможности предоставления услуг с использования биометрических данных

## Интеграция с ЕСИА

Интеграция с ЕСИА, реализована согласно актуальной версии «Методических рекомендаций по использованию Единой системы идентификации и аутентификации».

Для корректного взаимодействия требуется:

* доступ Системы к ЕСИА через сеть Интернет;
* Система должна быть зарегистрирована на техническом портале ЕСИА

## Интеграция с ЕБС

Для корректного взаимодействия с ЕБС требуется:

* корректно настроенное ПО "ТИБ ЕБС"
* ПО "ТИБ ЕБС" должно быть подключено к ЕБС через Крипто канал

## Зависимость от СМЭВ

Для корректного взаимодействия между ЕСИА, ЕБС и Системой:

* Система должна быть зарегистрирована в СМЭВ, через Личный кабинет участника взаимодействия, далее ЛКУВ
* для Системы должен быть выпущен сертификат Цифровой подписи:
  + для интеграционной среды сертификат выпускается через ЛКУВ;
  + для продуктовой среды - через аккредитованные удостоверяющие центры.
* Открытая часть сертификата Цифровой подписи должны быть загружена на техническом портале ЕСИА

## Требование к хранению ЦП

В рамках обмена информацией с ЕСИА все запросы должны быть подписаны сертификатом Цифровой подписью, далее ЦП. Для обеспечения корректного подписания запросов используется внешний КриптоПро DSS хранящий ЦП, расположенный в той же среде окружения. Хранение закрытой части и полной цепочки, а так же выпуск и учет сертификата ЦП накладывается на ответственного сотрудника Поставщика услуг

# [Системные требования](https://kb.akbars.ru/pages/viewpage.action?pageId=345181834)

Для развертывания Системы необходимы сервер приложения и сервер СУБД. Минимальные требования к этим серверам перечислены в следующей таблице.

| Параметр | Сервер приложения | Сервер приложения для работы с биометрическими образцами до 50 одновременных подключений | Сервер СУБД | Сервер с которого производится установка |
| --- | --- | --- | --- | --- |
| Количество ядер процессора, шт. | 4 | 24 | 2 | 1 |
| Тактовая частота процессора, не менее | 2,4 Ггц | 2,2 Ггц | 2,4 Ггц | 2,2 Ггц |
| Объем оперативной памяти, Гб | 8 | 24 | 8 | 2 |
| Стандарт и частота оперативной памяти, не менее | DDR4, 2.1 ГГц | DDR4, 2.1 ГГц | DDR4, 2.1 ГГц | DDR4, 2.1 ГГц |
| Объем логических дисков, Гб | 140 | 240 | 140 | 40 |
| Операционная система | Ubuntu 18.04 и выше  Astra Linux 1.7 и выше | Ubuntu 18.04 и выше  Astra Linux 1.7 и выше | Ubuntu 18.04 и выше  Astra Linux 1.7 и выше | Ubuntu 18.04 и выше  Astra Linux 1.7 и выше |
| Канал связи: внутренний доступ | 1 Гб/с | 1 Гб/с | 1 Гб/с | 100 Мб/с |
| Канал связи: внешний доступ | 100 Мб/с | 100 Мб/с | 100 Мб/с | 100 Мб/с |
| Программное обеспечение | Docker 20.10.9 и выше  Python 3.9 и выше | Docker 20.10.9 и выше  Python 3.9 и выше | Docker 20.10.9 и выше  Python 3.9 и выше | Docker 20.10.9 и выше  Python 3.9 и выше |

Для работы с Системой необходим клиентский компьютер, на котором должно быть установлено программное обеспечение, перечисленное в следующей таблице.

| Windows 7 и выше | Ubuntu 18.04 и выше |
| --- | --- |
| Браузер Google Chrome 72.0.3626.121 и выше (рекомендуется последняя версия) | Браузер Chromium 80 и выше |

## Системные требования

### Системные требования для серверов БД

* Процессор: 2 физических ядер.
* Объем оперативной памяти 8 Гб.
* Дисковая емкость: 1000Gb.
* Сетевой интерфейс: 1000 Мбит/с.
* Docker 20.10.9 и выше.

### Системные требования для серверов приложений и модулей Системы

* Процессор: 4 физических ядра (VCPU).
* Объем оперативной памяти 16 Гб.
* Дисковая емкость: 200Gb.
* Сетевой интерфейс: 1000 Мбит/с.
* Python 3.9
* Docker 20.10.9 и выше.

### Системные требования для запуска инсталлятора

* Операционная система Linux: Ubuntu 20.04 LTS и выше.
* Python 3.9
* Ansible 2.12

### Служебные учетные записи

Для развертывания релиза принято использовать следующие сервисные учетные записи:

* postgres: сервисная учетная запись базы данных PostgreSQL, являющаяся администратором кластера PostgreSQL;
* f2\_kbs: сервисная учетная запись базы данных PostgreSQL, являющаяся баз данных Системы в кластере PostgreSQL;
* f2\_kbs: служебная учетная запись для [RabbitMQ](https://www.rabbitmq.com/documentation.html), которую будут использовать ССП и СПО;

# [Варианты развертывания ПО Face2](https://kb.akbars.ru/pages/viewpage.action?pageId=345179069)

Система использует технологию Docker-контейнеризации. Cистема распространяется в виде готовых Docker образов собранных в защищенном контуре разработчика Системы. Образы Системы проверены на уязвимости с помощью статического и динамического анализатора кода.

### Варианты развертывания

Система устанавливается двумя способами. Выбор способа установки осуществляется администратором Системы. Способы установки:

* Docker: на выделенные серверы, с использованием Docker;
* Kubernetes: в систему оркестрации контейнеризированных приложений на базе Kubernetes.

Для каждого способа установки предназначен свой дистрибутив. Система поставляется с двумя дистрибутивами:

* в формате Helm: для установки в среду Kubernates
* в формате ansible также Инсталлятор: для установки на выделенные серверы.

Для дистрибутива доступен вариант развёртывания Системы в изолированной среде для демонстрации возможностей Системы.

* При развёртывании в среде демонстрации возможностей Системы все роли объединены на одном сервере.
* При развёртывании Системы в эксплуатационной среде все роли разделены между разными серверами.

### Роли серверов

Для каждого способа установки требуется определить роли для серверов. Для установки в Kubernetes:

* Кластер Kubernetes: кластер Kubernetes, на котором запущены микросервисы Системы.
* Сервер Инфраструктуры: сервер, на котором запущены вспомогательные инфраструктурные сервисы, необходимые для работы Системы. Для каждого инфраструктурного сервиса можно использовать отдельный сервер или кластер Kubernetes.
* Сервер Установки: сервер, с которого производится установка Системы.

### Для установки на выделенные серверы:

* Сервер БД: сервер, на котором запущен СУБД PostgreSQL, а также прочие инфраструктурные сервисы, не входящие в состав Системы, но используемые ей.
* Сервер Приложений: сервер, на котором запущены микросервисы Системы.
* Ansible мастер: сервер, с которого производится установка и/или обновление Системы.

# [Развертывание для демонстрации возможностей Системы](https://kb.akbars.ru/pages/viewpage.action?pageId=345179074)

Для демонстрации возможностей системы предоставляется готовый образ Виртуальной машины в которой развернуты все компоненты и модули Системы.

## Подготовка образа:

* Скачайте архив с образом ВМ (system\_vm.tgz) на ваш компьютер по предоставленной  ссылке.
* Распакуйте архив командой: tar -xzvf kbs\_face2\_vm.tgz -C /opt/face2/vm\_folder  
  Где */opt/face2/vm\_folder* - целевая директория для размещения ВМ (рекомендуется не менее 300 ГБ свободного места)

Запуск в VMware Player:

* Откройте VMware Player
* Выберите File → Open и укажите путь к файлу \*.vmx из распакованной директории

Настройте параметры ВМ (минимальные):

* Процессор: 8 физических ядер.
* Объем оперативной памяти 24 Гб.
* Дисковая емкость: 300Gb.
* Сетевой интерфейс: 1000 Мбит/с

Запустите ВМ кнопкой "Power On"

При старте система автоматически:

* Проверит целостность компонентов
* Инициализирует сервисы
* Настроит сетевые интерфейсы
* Учетные данные по умолчанию:
  + Логин: ubuntu
  + Пароль: ubuntu

После загрузки проверьте IP-адрес командой: ip a

## Важные примечания:

Образ содержит предустановленные компоненты:

* Docker с демо-контейнерами
* Web-интерфейс управления
* Личный кабинет пользователя

# [Развертывание ПО Face2 в Kubernetes](https://kb.akbars.ru/pages/viewpage.action?pageId=345179091)

Чтобы развернуть Систему Face2 в Kubernetes используйте инструкцию ниже.

## Подготовка среды для развертывания

Чтобы развернуть Систему в Kubernetes подготовьте среду, в которой будет происходить развертывание Системы.

Следует подготовить несколько серверов со следующими ролями:

| **Роль** | **Кол-во серверов** | **Функции** | **Программные требования** | **Аппаратные требования** |
| --- | --- | --- | --- | --- |
| **Мастер** | 1+ | Master kubernetes, ingress | <https://deckhouse.ru/gs/bm/step2.html> | <https://deckhouse.ru/gs/bm/step2.html> |
| **Нода** | 2+ | Worker+storage node kubernetes | как у роли **Мастер** | 16 ядер ЦП, 16 Гб ОЗУ, 200 Гб диск |
| **Установщик** | 1 | Сервер, с которого осуществляется установка | Ubuntu 18.04, 20.04, 22.04 | 2 ядер ЦП, 4 Гб ОЗУ, 100 Гб диск |

В сети должно быть настроено разрешение имён этих серверов в ip-адреса.

На серверах с ролью **Нода** должен существовать второй диск (/dev/sdb) объёмом 100 Гб.

**Важно!** Инструкция подразумевает наличие установленного и настроенного Deckhouse

## Подготовка дистрибутива

Подготовьте дистрибутив для сервера с ролью **Установщик**:

* Повысьте права командой sudo -i .
* Скопируйте полученный дистрибутив Системы на сервер с ролью Установщик (Далее - просто Установщик) в директорию /opt.
* Перейдите в директорию /opt командой cd /opt .
* Разархивируйте полученный дистрибутив Системы командой tar xzvf face2md.\*.tar.gz .
* Перейдите в директорию /opt/face2/scripts/ командой cd /opt/face2/scripts/ .
* Добавьте права на запуск файлов командой chmod +x \*.sh .

## Загрузка docker-образов в хранилище

### **Загрузка docker-образов в хранилище сервера с ролью Установщик**

Чтобы загрузить docker-образы в хранилище сервера с ролью **Установщик**:

* повысьте права командой sudo -i .
* Перейдите в директорию /opt/face2/scripts/ командой cd /opt/face2/scripts/ .
* Запустите скрипт подготовки системы, который установит или обновит Docker и прочие необходимые пакеты следующей командой :
  + ./install\_docker\_Ubuntu.sh
* Перезагрузите систему в конце работы скрипта, ответьте **Y**, чтобы перезагрузить операционную систему.
* Повысьте права после перезагрузки командой sudo -i .
* Перейдите в директорию docker в директории дистрибутива командой cd /opt/face2/docker .
* Загрузите docker-образы в локальное хранилище командами:
  + docker load --input face2\_ebsidp.tar ;
  + docker load --input face2\_kbs\_auth.tar ;
  + docker load --input face2\_kbs\_payments.tar ;
  + docker load --input face2\_ageconfirmation.tar ;
  + docker load --input face2\_kbs\_adminpanel.tar ;
  + docker load --input face2\_triton.tar ;
* Определите, в какой сетевой репозиторий будут загружаться docker-образы дистрибутива Системы. Если сетевой репозиторий не существует или требуется создать новый:
  + создайте директории для сертификатов, используемых docker registry, командой mkdir -p /opt/registry/certs .
  + Поместите в директорию /opt/registry/certs сертификат сервера  с ролью **Установщик** и его ключ. Сертификат должен содержать всю цепочку удостоверяющих центров, которым в свою очередь должны доверять все серверы кластера.
  + Для создания директории для хранения docker-образов используйте команду mkdir -p /opt/registry/storage .
  + Для установки на сервере с ролью **Установщик** сетевого репозитория docker-образов Docker Registry, работающего по протоколу https на порте 5000, выполните

docker run -d \

  -p 5000:5000 \

--restart=unless-stopped \

  --name registry \

  -v /opt/registry/storage:/var/lib/registry \

  -v /opt/registry/certs:/certs \

  -e REGISTRY\_HTTP\_TLS\_CERTIFICATE=/certs/domain.crt \

  -e REGISTRY\_HTTP\_TLS\_KEY=/certs/domain.key \

  registry:2

* где:
  + domain.crt :имя файла сертификата сервера,
  + domain.key :имя файла с ключом сертификата.
* После выполнения команды на сервере с ролью **Установщик** будет запущен контейнер с сетевым хранилищем docker-образов Docker Registry, работающий на порте 5000 по протоколу https и персистентным хранилищем в директории /opt/registry/storage .
* Измените теги для всех docker-образов дистрибутива Системы. Вместо репозитория face2 в тэге образа пропишите тэг текущего сетевого репозитория docker-образов, используя скрипт из дистрибутива:
  + Перейдите в директорию /opt/face2/scripts/ командой cd /opt/face2/scripts/ .
  + Выполните скрипт sh командой ./dockerRetagAndPush.sh <имяРепозитория> , в поле <имяРепозитория> укажите имя своего сетевого репозитория docker-образов. Если используется Docker Registry, запущенный на сервере на порт 5000, укажите localhost:5000. Скрипт заменит имя репозитория в тэгах docker-образов и загрузит их в сетевое хранилище.

## Установка утилит kubectl и helm

### Установка утилит kubectl и helm на сервере с ролью Установщик

Чтобы установить утилиты kubectl и helm на сервере с ролью **Установщик**

* Установите утилиту kubectl. Выполните команды: curl -LO "<https://dl.Kubernetes.io/release/$>(curl -L -s <https://dl.Kubernetes.io/release/stable.txt>)/bin/linux/amd64/kubectl"; install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl; kubectl version --client.
* Установите утилиту helm. Выполните команды: curl -fsSL -o get\_helm.sh <https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3> chmod 700 get\_helm.sh ./get\_helm.sh helm version
* Настройте kubectl на работу с кластером Kubernetes. Для настройки перейдите в браузере на страницу по адресу https://kubeconfig.<доменное имя кластера Kubernetes>, например, k8dev.md.local и следуйте инструкциям в разделе Linux.
* Проверьте работу kubectl, выполнив команду kubectl get namespaces. При корректной работе команды на экран будет выведен список неймспейсов кластера Deckhouse.

### Конфигурация параметров стенда перед установкой

Конфигурация параметров стенда перед установкой на сервере с ролью **Установщик**:

* Перейдите в директорию /opt/face2/helm в директории дистрибутива командой cd /opt/face2/helm.
* Скопируйте файл stand/stand-template.yaml в директорию stand командой cp stand/stand-template.yaml stand/<имя нового стенда>.yaml. Этот файл содержит параметры стенда, на который будет установлен экземпляр Системы.
* Отредактируйте файл stand/<имя нового стенда>.yaml, актуализировав все необходимые параметры, следуя тексту в комментариях.
* Отредактируйте в файле templates/secrets/postgres-admin-secret.yaml параметр **data.password**, заменив значение SuperSecret123@ на свой пароль роли postgres в кластере Postgres.
* Отредактируйте файлы values.yaml и stand/3.2.0.yaml, актуализировав все необходимые параметры, следуя тексту в комментариях и заменив registry: face2 на registry: <Адрес сетевого репозитория docker-образов, содержащего docker-образы данного дистрибутива Системы>. При использовании Docker Registry можно указать <FQDN сервера с ролью Установщик>:5000. Для массовой замены значений в файле используйте команду sed -i -e "s/<что заменить>/<на что заменить>/g" <путь до файла>.

Установка Системы

Установка Системы на сервере с ролью Установщик

Чтобы установить Систему на сервере с ролью Установщик:

* Перейдите в корень директории дистрибутива командой cd /opt/face2/helm.
* Запустите установку Системы командой

helm upgrade \

--install \

--wait \

--timeout 2000s \

-n <имя неймспейса Kubernetes> \

--create-namespace \

--values stand/<имя стенда>.yaml . \

--debug > logs.txt

* где:
  + имя неймспейса Kubernetes: пространство имен Kubernetes, например, test.
  + имя стенда: имя стенда, на котором устанавливается Система, например, test.
* Дождитесь сообщения об успешной установке.

# [Проверка работоспособности ПО Face2](https://kb.akbars.ru/pages/viewpage.action?pageId=345180110)

Чтобы проверить работоспособность Системы:

* Перейдите по ссылке в браузере https://<имя стенда>.<внешний домен кластера Kubernetes>/admin/, где:
  + имя стенда: имя стенда, на котором устанавливается Платформа, например, test.
  + внешний домен кластера Kubernetes: доменный суффикс кластера Kubernetes, например, *[Kubernetes.md](http://kubernetes.md/)*.local.
* Найдите с помощью поиска те развёртывания, у которых плитки одноимённых сервисов в административной консоли Платформы окрашены красным, и перезапустите их.
* Запустите установку Helm ещё раз.
* Проверьте доступность всех ресурсов Системы.

## **Просмотр статуса здоровья для каждого сервиса и модуля**

Во всех модулях Системы встроены проверки здоровья и предоставляют информацию по доступности всех зависимых ресурсов Системы.

Для проверки зависимых ресурсов системы необходимо перейти по адресу проверки ресурсов в браузере https://<адрес сервиса>.<внешний домен кластера Kubernetes>/<адрес проверки здоровья>/   , где:

* адрес сервиса: адрес сервиса или модуля к которому производится проверка;
* внешний домен кластера Kubernetes: доменный суффикс кластера Kubernetes, например, *[Kubernetes.md](http://kubernetes.md/)*.local;
* адрес проверки здоровья: адрес простой или полной проверки здоровья.

## **Настроенные проверки здоровья**

NpgSqlHealthCheck  - проверка подключения к базе данных PostgreSQL. Использует строку подключения к БД из конфигурации

EbsUrlCheck  - проверка доступности URL EBS. URL берется из конфигурации Ebs:Url

EsiaUrlCheck  - проверка доступности URL ЕСИА. URL берется из конфигурации Esia:Url

CryptoProUrlCheck  - проверка доступности URL КриптоПро DSS. URL берется из конфигурации CryptoPro:Dss:Url

AsopUrlCheck  - проверка доступности URL АСОП. URL берется из конфигурации Asop:Url

RabbitMQ  - проверка подключения к RabbitMQ. Использует строку подключения из конфигурации RabbitMq

#### **Использование Health Checks**

Адрес простой проверки ресурсов (health/C2AOXALCK4/)

* Возвращает статус 200 OK, если приложение запущено (все проверки отключены)
* Использовать для простой проверки, что приложение запущено и отвечает

Адрес полной проверки ресурсов (health/full/C2AOXALCK4/)

* Выполняет все проверки здоровья и возвращает детализированный JSON-отчет
* Использовать для полной диагностики состояния приложения и его зависимостей

## **Интерпретация результатов**

Коды статусов HTTP

* 200 OK: Все проверки прошли успешно (Healthy)
* 500 Internal Server Error: Обнаружены проблемы в работе связанных подсистем (Degraded)
* 503 Service Unavailable: Критические проверки не прошли (Unhealthy)