

Общество с ограниченной ответственностью «Европейская Юридическая Служба»
(ООО «ЕЮС»)

СПЕЦИАЛИЗИРОВАННОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
«ДОБРОСЕРВИС»

ИНСТРУКЦИЯ ПО УСТАНОВКЕ
На 33 листах

Москва, 2025

СОДЕРЖАНИЕ

1 ОБЩИЕ СВЕДЕНИЯ.....	2
1.1 Полное наименование системы и ее условное обозначение.....	2
1.2 Назначение и цели создания системы.....	2
1.2.1 Назначение системы	2
1.2.2 Цели создания системы	2
1.3 Термины и сокращения	3
2 УСТАНОВКА (ПОДГОТОВКА К РАБОТЕ).....	10
2.1 Настройка VPN-сервиса	10
2.2 Доступ к серверам через SSH-подключение	12
2.2.1 Описание сервисов	15
2.2.1.1 API (основной backend)	15
2.2.1.2 Auth (Сервис аутентификации)	17
2.2.1.3 Gateway (API Gateway)	19
2.2.1.4 Phone (Телефония).....	20
2.2.1.5 Files Storage (Файловое хранилище)	22
2.2.1.6 Chat (Чат-сервис)	23
2.2.1.7 CRM (Система управления клиентами).....	25
2.2.1.8 Личный кабинет	28
2.2.2 Инфраструктурные сервисы.....	29
2.2.3 Мониторинг и наблюдаемость	29
2.2.4 Команды для быстрого доступа к контейнерам	30
2.2.5 Полезные команды внутри контейнеров	31
3 КОНТАКТЫ ТЕХНИЧЕСКИХ СПЕЦИАЛИСТОВ	32

1 ОБЩИЕ СВЕДЕНИЯ

1.1 Полное наименование системы и ее условное обозначение

Полное наименование: специализированное программное обеспечение Добросервис.

Краткое наименование: система, информационная система, СПО, Добросервис, платформа.

1.2 Назначение и цели создания системы

1.2.1 Назначение системы

СПО Добросервис – программа круглосуточной информационно-психологической поддержки персонала, комплекс решений для укрепления ментального здоровья и благополучия.

Задачи программы:

- Решение личных и профессиональных проблем, влияющих на продуктивность труда и мотивацию работников;
- Повышение ответственности за укрепление собственного здоровья сотрудника, психологического состояния;
- Поддержка эффективности и работоспособности сотрудников;
- Предоставление информационной помощи по направлениям: психологическое здоровье, юридическая поддержка, финансовая консультация, телемедицина и ЗОЖ, критические инциденты;
- Повышение удовлетворенности сотрудников работой и улучшение их психологического состояния;
- Предоставление статистической отчетности HR-отделу для оценки качества инвестиций в программы поддержки сотрудников;
- Обеспечение конфиденциальности оказываемых услуг.

1.2.2 Цели создания системы

Целями разработки специализированного программного обеспечения Добросервис являются:

- **Повышение эффективности бизнеса.** Улучшение мотивации и снижение факторов стресса, негативно влияющих на рабочую активность сотрудников.
- **Укрепление корпоративной культуры.** Формирование доверительной атмосферы, способствующей открытому общению и решению внутренних конфликтов.

- **Предупреждение профессионального выгорания.** Своевременная диагностика и профилактика эмоционального истощения среди сотрудников.
- **Обучение управлению стрессовыми ситуациями.** Развитие навыков саморегуляции и повышения устойчивости к внешним негативным факторам.
- **Развитие социальной ответственности компании.** Демонстрация заботы о сотрудниках, улучшение имиджа работодателя на рынке труда.
- **Минимизация рисков ухода ценных кадров.** Предотвращение негативных последствий стресса и неудовлетворённости трудом через своевременную психологическую поддержку.
- **Оптимизация управления персоналом.** Использование полученной обратной связи для улучшения процессов подбора, адаптации и развития сотрудников.
- **Психологическая подготовка сотрудников к изменениям.** Поддержка адаптации к нововведениям внутри организации, минимизация сопротивления переменам.
- **Создание условий для карьерного роста.** Консультации и поддержка сотрудников в процессе личностного и профессионального развития.

1.3 Термины и сокращения

№ п.п.	Термин или сокращение	Расшифровка
1.	Авторизация	Процесс подтверждения прав доступа к функционалу Системы.
2.	Администратор	Человек или группа людей, которые эксплуатируют информационную систему
3.	АРМ	Автоматизированное рабочее место
4.	БД	База данных.
5.	Бизнес-процесс	Процесс обработки данных, для которого задана точка входа и несколько точек выхода и последовательность действий (шагов, этапов, функций), совершаемых в заданном порядке и в определенных условиях.
6.	Время отклика или реакция системы, response time	Временной интервал между началом инициации информации и моментом, когда соответствующий

№ п.п.	Термин или сокращение	Расшифровка
		отклик становится доступным в определенных условиях (ГОСТ Р МЭК 61069-4-2012).
7.	ГОСТ	Государственный стандарт
8.	ГОСТ Р	Государственные стандарты Российской Федерации
9.	Графический интерфейс пользователя, Graphical user interface, GUI	Пользовательский интерфейс, в котором элементы (меню, кнопки, значки, списки и т. п.), представленные пользователю на экране, исполнены в виде графических изображений.
10.	Данные	Информация, представленная в формализованном виде, пригодном для передачи, интерпретации или обработки с участием человека или автоматическими средствами.
11.	Данные о сотрудниках	Личная информация, сведения о работе и оценки эффективности отдельных сотрудников.
12.	Датасет	Коллекция из логических записей, хранящихся в виде кортежа
13.	Задачи	Определенные рабочие элементы, назначенные отдельным сотрудникам или командам.
14.	ЗИ	Защита информации.
15.	Информационная система (ИС)	Информационная система, являющаяся предметом настоящего технического задания, включает все компоненты, предполагаемые к поставке и их интеграции.
16.	Мониторинг в реальном времени	Непрерывный мониторинг вызовов и производительности системы, с целью подтверждения корректной работы системы, а также выявления проблем.
17.	Обработка естественного языка	Область исследования, направленная на то, чтобы позволить компьютерам понимать и обрабатывать человеческий язык.

№ п.п.	Термин или сокращение	Расшифровка
18.	Облачная инфраструктура	Модель доставки вычислительных ресурсов через Интернет, обеспечивающая масштабируемость и доступность.
19.	ОПО	Общее программное обеспечение
20.	ОС	Операционная система
21.	Отчёт	Сводки ключевых показателей производительности и других данных, генерируемых системой.
22.	ПАК	Программно-аппаратный комплекс
23.	Пиковая производительность	Производительность, рассчитанная с использованием базовой частоты процессора (специфицированной Производителем) в вычислениях над 64 разрядными операндами с плавающей запятой
24.	ПК	Персональный компьютер.
25.	ПО	Программное обеспечение.
26.	Подсистема	Часть информационной системы, обладающая свойствами системы, и в частности, имеющая подцель.
27.	Поисковая система	Программная система, которая ищет и извлекает информацию из базы данных на основе пользовательских запросов.
28.	Проект	Перечень работ, выполняемых в рамках Договора.
29.	Процессор	Устройство (процессорный чип), содержащее несколько процессорных ядер
30.	Рабочий процесс	Последовательности задач и процессов, разработанных для достижения конкретных бизнес-целей.
31.	РД	Руководящий документ
32.	Роль	Совокупность прав на действия с сущностями и настройками Системы.

№ п.п.	Термин или сокращение	Расшифровка
33.	Сайзинг	Подбор оптимальной конфигурации аппаратного обеспечения для какой-либо информационной системы.
34.	Свободное программное обеспечение, СПО	Программное обеспечение, распространяемое на условиях открытой лицензии, которые позволяют пользователю: — использовать программу для ЭВМ в любых, не запрещенных законом целях; — получать доступ к исходным текстам (кодам) программы как в целях изучения и адаптации, так и в целях переработки программы для ЭВМ; распространять программу (бесплатно или за плату, по своему усмотрению); — вносить изменения в программу для ЭВМ (перерабатывать) и распространять экземпляры измененной (переработанной) программы с учетом возможных требований наследования лицензии; — в отдельных случаях распространять модифицированную компьютерную программу пользователем на условиях, идентичных тем, на которых ему предоставлена исходная программа. (ГОСТ Р 54593-2011).
35.	Семантический поиск	Метод поиска, который использует алгоритмы машинного обучения для понимания значения и контекста запроса пользователя и возвращает релевантные результаты на основе этого понимания.
36.	Система управления базами данных, СУБД	Совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.
37.	СПО	Специализированное программное обеспечение

№ п.п.	Термин или сокращение	Расшифровка
38.	СУБД	Система управления базами данных
39.	СХД	Система хранения данных
40.	Телемедицина	Использование информационно-коммуникационных технологий для предоставления медицинских услуг на расстоянии
41.	Технологический стек	Набор используемых технологий (платформ, языков программирования и т.д.).
42.	Транскрипция	Письменное воспроизведение слов и текстов с учётом их произношения средствами определённой графической системы
43.	ТС	Технические средства
44.	ЦОД	Дата-центр (от англ. data center), или центр (хранения и) обработки данных (ЦОД/ЦХОД) — это специализированное здание для размещения (хостинга) серверного и сетевого оборудования и подключения абонентов к каналам сети Интернет.
45.	Частная телефонная станция (PBX)	Телефонная система, используемая в бизнесе, которая позволяет совершать и принимать звонки как внутри, так и снаружи.
46.	Application programming interface, API	Описание способов (набор классов, процедур, функций, структур или констант), которыми приложение может взаимодействовать с другим.
47.	Asterisk	Виртуальная автоматическая телефонная станция
48.	Continuous Integration/Continuous Delivery, CI/CD	Непрерывная интеграция и доставка. Практика непрерывного слияния рабочих копий программного кода в общую основную ветвь, с последующей доставкой и развертыванием завершённого кода, включая различные типы автоматических тестов на каждом этапе.
49.	Docker	Программное обеспечение для автоматизации развертывания приложения. Позволяет «упаковать» приложение со всем его окружением в контейнер,

№ п.п.	Термин или сокращение	Расшифровка
		который может быть перенесен на любую Linux-систему. Лицензия свободного ПО https://www.docker.com/legal/components-licenses
50.	ETL (Extract, Transform, Load — дословно «извлечение, преобразование, загрузка»)	Процесс в управлении данными, включающий в себя извлечение данных из внешних источников; их трансформация и очистка, чтобы они соответствовали потребностям бизнес-модели; и загрузка их в базу данных.
51.	HTTP	HyperText Transfer Protocol – протокол передачи гипертекстовых файлов
52.	HTTPS	HyperText Transfer Protocol Secure - расширение протокола HTTP, поддерживающее шифрование
53.	Message Queues, MQ	ПО, которое обеспечивает гарантированную доставку сообщений между системами на разных платформах.
54.	Representational State Transfer API, REST API	Архитектурный стиль взаимодействия компонентов распределённого приложения в сети.
55.	Simple Object Access Protocol, SOAP	Протокол обмена структурированными сообщениями в распределённой вычислительной среде.
56.	Single Sign-On, SSO	Технология, при использовании которой пользователь переходит из одного раздела системы в другой без повторной аутентификации.
57.	SSH (Secure Shell)	Сетевой протокол для безопасного удалённого управления серверами и устройствами. Разработан для защиты данных и предотвращения несанкционированного доступа.
58.	Standalone – приложение, Standalone	Это программное обеспечение (ПО), которое не нуждается в каких-либо дополнительных

№ п.п.	Термин или сокращение	Расшифровка
		программах и зависимостях для его установки и функционирования.

2 УСТАНОВКА (ПОДГОТОВКА К РАБОТЕ)

2.1 Настройка VPN-сервиса

1. Для начала работы необходимо скачать установочный файл приложения Cisco для ПК, для этого перейдите по ссылке <https://softvpn.els24.com>

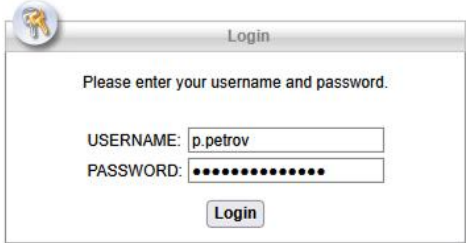
2. Необходимо пройти процесс идентификации с помощью формы **Login**:

2.1. Необходимо указать *Username* пользователя.

Username	p.petrov
----------	----------

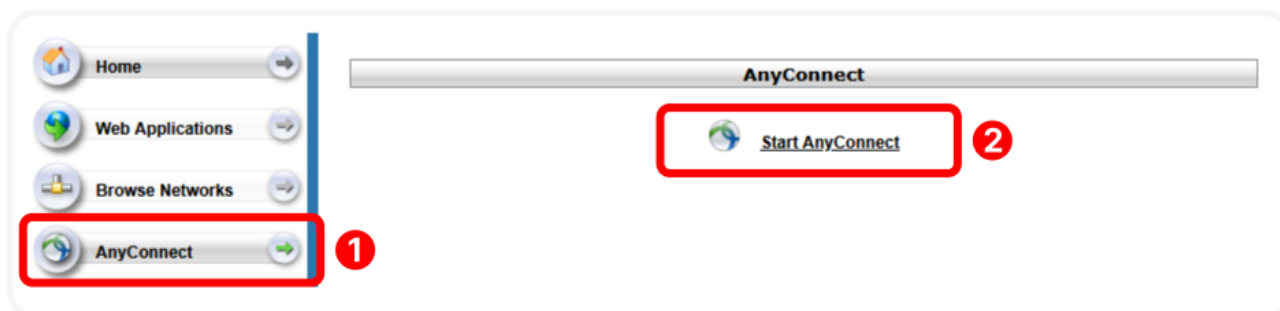
2.2. Затем указать *пароль* пользователя

Password	8h\$@Xp~ZxwJrvA
----------	-----------------

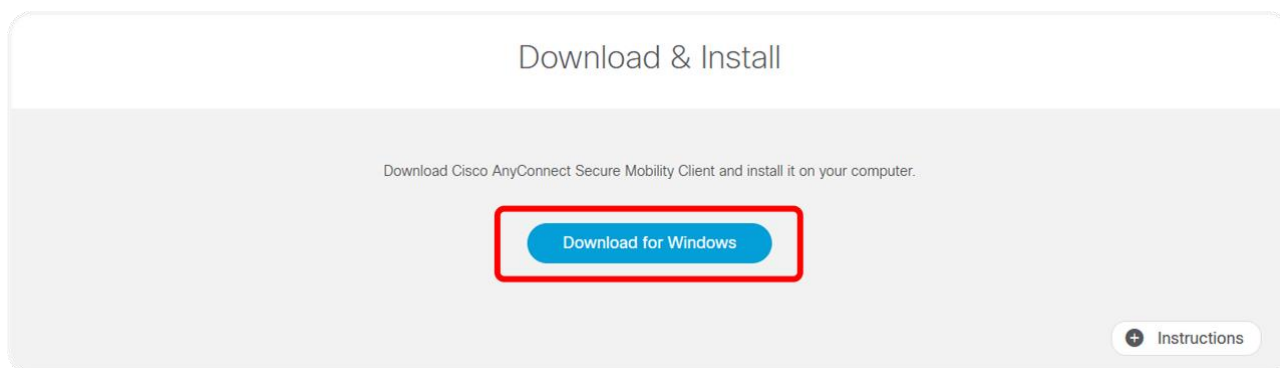


2.3. Далее нажать кнопку **Login**

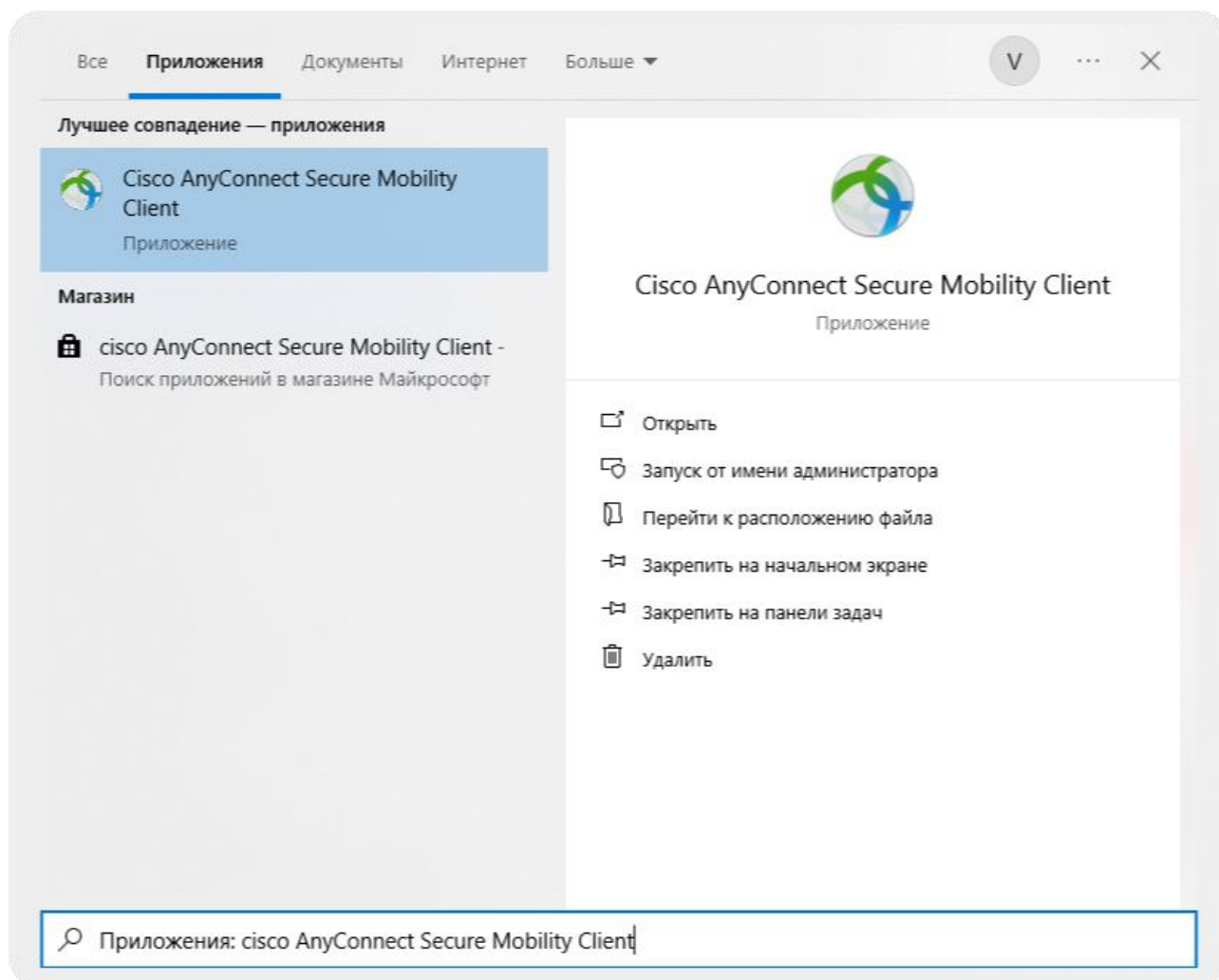
3. На открывшейся странице в навигационном меню выбрать пункт меню **AnyConnect**, затем нажать **Start AnyConnect**.



4. Затем нажмите **Download for Windows** и установите скаченный клиент.



5. После установки найдите и запустите установленную программу **Cisco AnyConnect Secure Mobility Client**

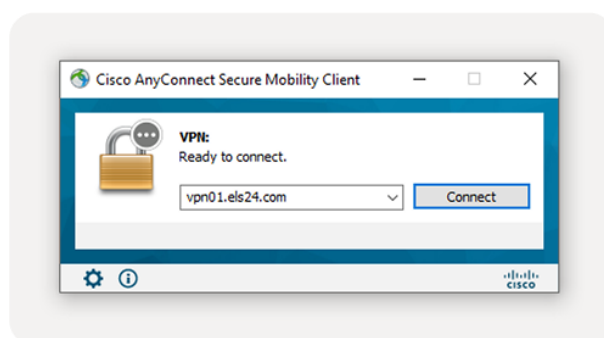


6. Введите данные сервера для подключения

6.1 Указать *Сервер*

Сервер	vpn01.els24.com
--------	-----------------

6.2. Затем нажать кнопку **Connect**



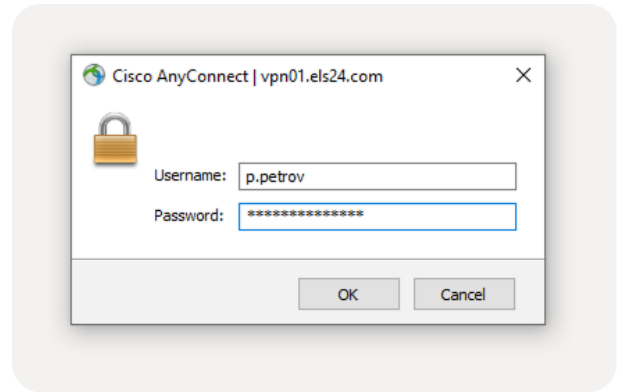
6.3 Далее, необходимо указать

Username пользователя.

Username	p.petrov
----------	----------

6.4. Затем указать *пароль* пользователя

Password	8h\$@Xp~ZxwJrvA
----------	-----------------



6.3. Нажать кнопку **Ok**

2.2 Доступ к серверам через SSH-подключение

1. Для начала работы необходимо скачать установочный файл приложения PuTTY для ПК, для этого перейдите на официальный сайт разработчика по ссылке <https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

2. Загрузите подходящую версию приложения под разрядность операционной системы.

Package files

You probably want one of these. They include versions of all the PuTTY utilities (except the new and slightly experimental Windows pterm).

(Not sure whether you want the 32-bit or the 64-bit version? Read the [FAQ entry](#).)

We also publish the latest PuTTY installers for all Windows architectures as a free-of-charge download at the [Microsoft Store](#); they usually take a few days to appear there after we release them.

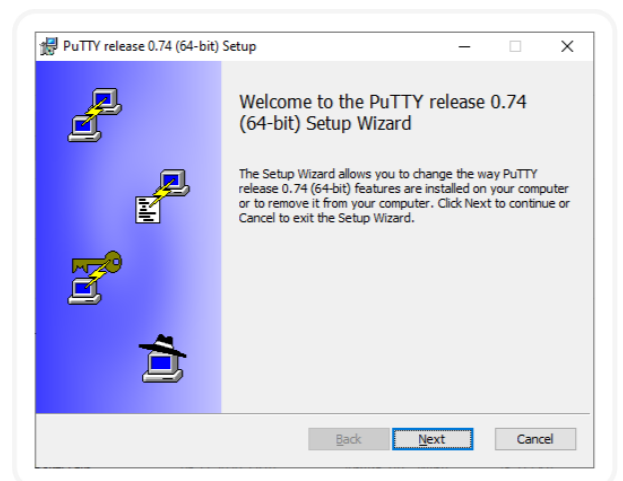
MSI ("Windows Installer")

64-bit x86:	putty-64bit-0.83-installer.msi	(signature)
64-bit Arm:	putty-arm64-0.83-installer.msi	(signature)
32-bit x86:	putty-0.83-installer.msi	(signature)

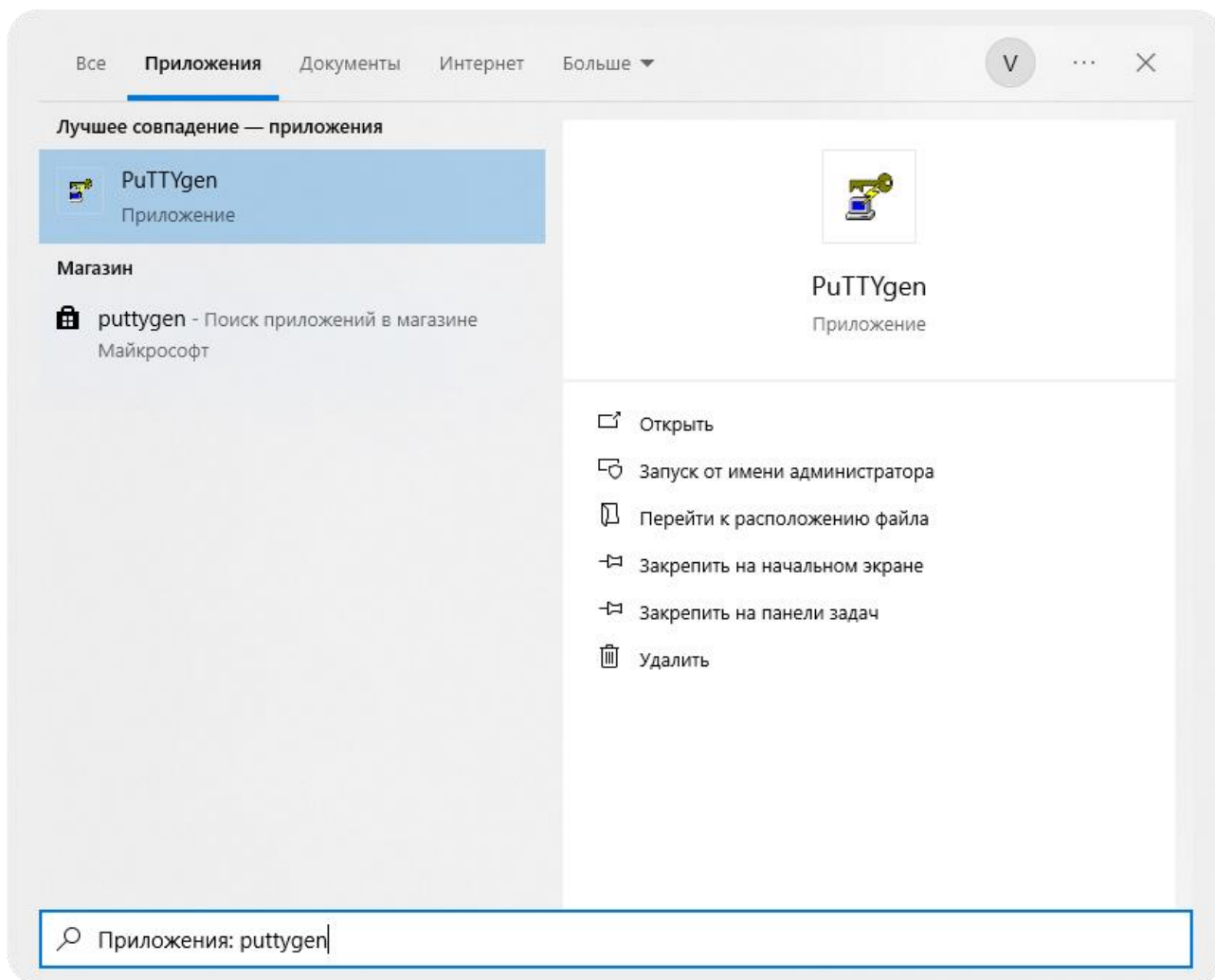
Unix source archive

.tar.gz:	putty-0.83.tar.gz	(signature)
----------	-----------------------------------	-----------------------------

3. Скачайте инсталлятор и запустите его – отобразится окно приветствия, кликаем по кнопке «Next» до конечного пункта.



4. После установки найдите и запустите установленную программу **PuTTY**



5. Доступ к серверам осуществляется по SSH-протоколу. Для каждого сервера используется краткий синоним, который применяется далее по всему разделу. Для получения доступа к node 1 (основной сервер с CRM, Backend API, Gateway, телефонией, файловым хранилищем) необходимо указать IP-адрес сервера, логин и пароль.

5.1. На вкладке Session в поле *Host Name (or IP address)* необходимо
вписать IP

IP address	10.88.5.94
------------	------------

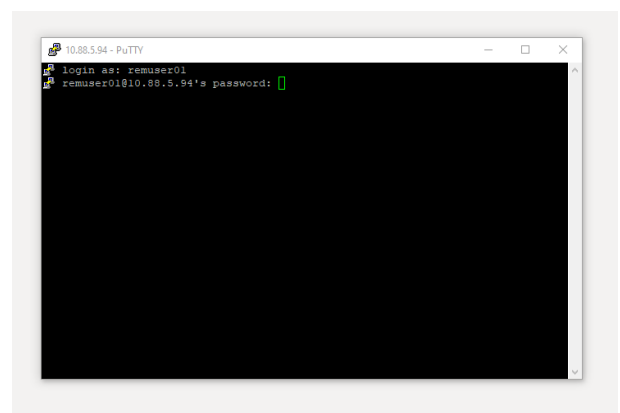
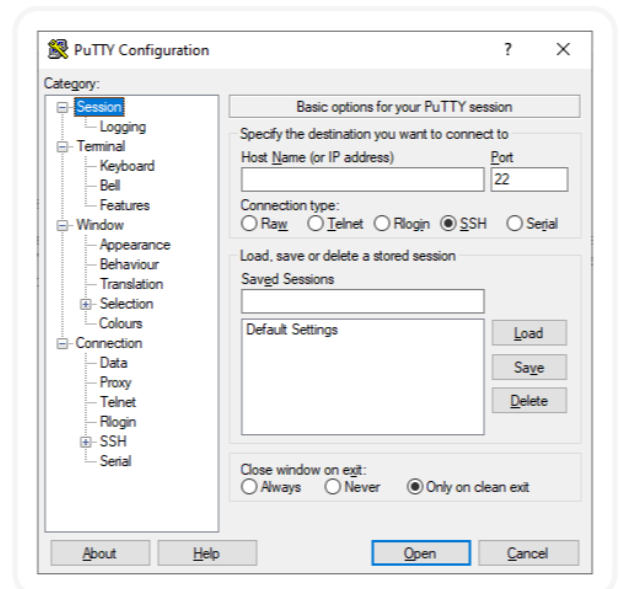
5.2. Затем нажать кнопку **Open**

5.3. Далее, необходимо указать *Login*
пользователя.

Login as	remuser01
----------	-----------

5.4. Затем указать *пароль* пользователя

Password	Rebred0olRix%
----------	---------------

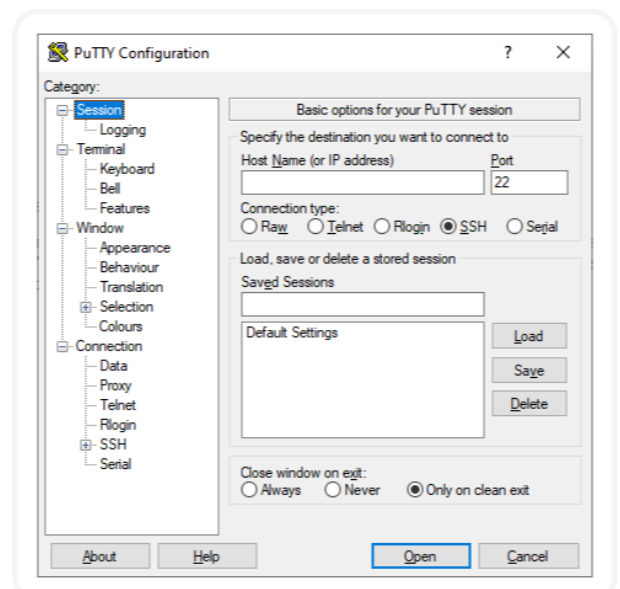


6. Для получения доступа к node 2 (сервер с Chat-сервисом и Личными кабинетами)
необходимо указать IP-адрес сервера, логин и пароль.

6.1. На вкладке Session в поле *Host Name (or IP address)* необходимо
вписать IP

IP address	10.0.8.41
------------	-----------

6.2. Затем нажать кнопку **Open**



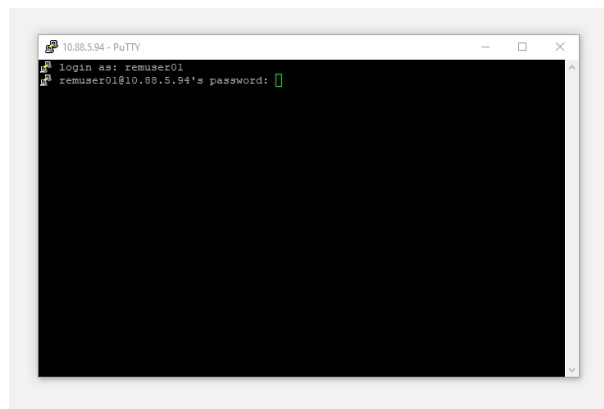
6.3. Далее, необходимо указать *Login*

пользователя.

Login as	remuser01
----------	-----------

6.4. Затем указать *пароль* пользователя

Password	Rubid4omnag)
----------	--------------



2.2.1 Описание сервисов

2.2.1.1 API (основной backend)

Основной бэкенд-сервис, реализующий бизнес-логику платформы. Обрабатывает запросы от CRM, ЛК, ЛК HR и мобильного приложения. Работает с базой данных, внешними API, уведомлениями, телефонией.

Хост:

node1

Контейнер:

Имя: deploy-api-dev-1

Команда для входа: `docker exec -it deploy-api-dev-1 /bin/sh`

Расположение файлов:

/var/www/html/	- корневая директория приложения (Laravel 8)
├── app/	- код приложения (модели, контроллеры, сервисы)
│ ├── App/	- основной код приложения
│ │ ├── Console/	- консольные команды
│ │ ├── Exceptions/	- обработчики исключений
│ │ ├── Http/	- HTTP слой (Controllers, Middleware, Requests)
│ │ ├── Providers/	- сервис-провайдеры Laravel
│ │ └── Models/	- Eloquent модели
│ ├── Domain/	- доменная логика (DDD подход, 40+ доменов)
│ │ └── B2B/	- B2B бизнес-логика (Controllers, Middlewares,
│ ├── Processors, Services)	
│ │ ├── Cards/	- управление карточками клиентов (Card, CardClient,
│ │ ├── CardService, CardStatus)	
│ │ ├── CLS/	- система юридических вопросов (CLSFile, CLSService,
│ │ ├── CLSSource, CLSTask)	
│ │ ├── Clients/	- управление клиентами (11 поддоменов: случаи,
│ │ ├── категории, поиск, импорт)	
│ │ ├── Companies/	- юридические лица (Company, CompanyInfo,
│ │ ├── CompanyLegalData, CompanySearcher)	
│ │ ├── Correspondence/	- корреспонденция и документооборот
│ │ ├── Tasks/	- задачи (17 поддоменов: типы, статусы, напоминания,
│ │ ├── история, теги)	
│ │ ├── Users/	- пользователи (15 поддоменов: роли, рейтинги,
│ │ ├── подписки, группы, отделы)	
│ │ └── Services/	- услуги платформы (категории, тарифы, подписки,
└── пакеты)	

сотрудники)	Statistics/	- статистика (40+ типов: звонки, клиенты, задачи,
	Notifications/	- уведомления (типы, шаблоны, история отправки)
	Documents/	- документы (генерация, шаблоны, электронная подпись)
	Payments/	- платежи (транзакции, счета, подписки, возвраты)
	Communications/	- коммуникации (email, SMS, чаты)
	Integrations/	- интеграции с внешними сервисами
	[другие 25+ доменов по различным бизнес-направлениям]	
	GraphQL/	- GraphQL resolvers
	Queries/	- GraphQL запросы
	Mutations/	- GraphQL мутации
	Types/	- типы GraphQL
	Scalars/	- кастомные скаляры
	Support/	- вспомогательные классы (helpers, traits, utilities)
	bootstrap/	- файлы инициализации Laravel
	config/	- конфигурационные файлы (56 файлов: app, auth,
database, services и др.)		
	database/	- миграции, seeders, factories
	migrations/	- миграции БД (история изменений схемы)
	seeders/	- начальные данные (справочники, тестовые данные)
	factories/	- фабрики для тестов
	packages/	- локальные пакеты (eus/social-auth и другие)
	eus/	- пространство имен EUS
	social-auth/	- OAuth авторизация через соцсети
	public/	- публичная директория (index.php - точка входа)
	resources/	- ресурсы (views, lang, assets)
	views/	- шаблоны Blade
	lang/	- локализация (ru, en, vn)
	assets/	- фронтенд ресурсы
	routes/	- маршруты API (api.php, web.php, channels.php)
	storage/	- хранилище (логи, кеш, загрузки)
	app/	- загруженные файлы (public, temp, uploads)
	framework/	- кеш, сессии, compiled views
	logs/	- логи приложения (laravel-dev.log)
	clockwork/	- debug данные (Clockwork профайлер)
	tests/	- тесты (Unit, Feature, Integration)
	Unit/	- юнит-тесты
	Feature/	- функциональные тесты
	vendor/	- зависимости Composer (Laravel, библиотеки)

Детализация основных доменов:

1. Domain/Tasks/ (управление задачами):

- Task, TaskComment, TaskFile, TaskHistory, TaskStatus;
- TaskReminder, TaskTemplate, TaskType, TaskPriority;
- TaskAssignee, TaskTag, TaskCategory, TaskFilter;
- Controllers, Services, Repositories, Requests, Resources.

2. Domain/Clients/ (управление клиентами):

- Client, ClientCategory, ClientCase, ClientContact;
- ClientImport, ClientSearch, ClientMerge, ClientHistory;
- ClientRetention, ClientSegment, ClientNote.

3. Domain/Statistics/ (аналитика):

- CallStatistics, ClientStatistics, TaskStatistics;
- UserStatistics, ServiceStatistics, PaymentStatistics;
- ReportBuilder, ChartGenerator, DataExporter.

Volumes (монтируемые директории):

- /volumes/api-dev/app → /var/www/html/storage/app
- /volumes/api-dev/logs → /var/www/html/storage/logs
- /volumes/api-dev/clockwork → /var/www/html/storage/clockwork

База данных:

Подключение к MySQL Galera Cluster: 10.88.5.254

База данных: eus_dev_api

Пользователь: eus_dev_api

2.2.1.2 Auth (Сервис аутентификации)

Сервис аутентификации и авторизации. Выдает JWT-токены, управляет пользовательскими сессиями, обрабатывает восстановление паролей, SMS-подтверждения, OAuth интеграции.

Хост:

node1

Контейнер:

Имя: deploy-auth-dev-1

Команда для входа: docker exec -it deploy-auth-dev-1 /bin/sh

Расположение файлов:

/var/www/html/	- корневая директория (Lumen 8 микрофреймворк)
├── app/	- код приложения
│ ├── Http/	- HTTP слой
│ │ ├── Controllers/	- контроллеры
│ │ │ ├── AuthController.php	- аутентификация (login, logout)
│ │ │ ├── RegisterController.php	- регистрация новых пользователей
│ │ │ ├── PasswordController.php	- восстановление пароля
│ │ │ ├── TokenController.php	- управление JWT токенами
│ │ │ ├── SmsController.php	- SMS подтверждение номера телефона
│ │ │ ├── OAuthController.php	- OAuth авторизация (Google, Facebook, VK)
│ │ │ └── SessionController.php	- управление сессиями
│ │ ├── Middleware/	- middleware
│ │ │ ├── Authenticate.php	- проверка аутентификации
│ │ │ ├── CheckJWT.php	- валидация JWT токенов
│ │ │ ├── RateLimiter.php	- ограничение частоты запросов
│ │ │ └── CorsMiddleware.php	- CORS заголовки
│ │ ├── Requests/	- валидация запросов
│ │ │ ├── LoginRequest.php	- валидация логина
│ │ │ ├── RegisterRequest.php	- валидация регистрации
│ │ │ └── PasswordResetRequest.php	- валидация сброса пароля
│ └── Services/	- бизнес-логика
│ ├── AuthService.php	- логика аутентификации
│ └── JwtService.php	- генерация и проверка JWT

— PasswordService.php	- работа с паролями (hash, verify)
— SmsService.php	- отправка SMS кодов
— OAuthService.php	- интеграция с OAuth провайдерами
— SessionService.php	- управление сессиями
— TokenRefreshService.php	- обновление токенов
— Repositories/	- работа с БД
— UserRepository.php	- пользователи
— TokenRepository.php	- токены (JWT, refresh, reset)
— SessionRepository.php	- сессии
— OAuthProviderRepository.php	- OAuth провайдеры
— Models/	- Eloquent модели
— User.php	- пользователь
— Token.php	- токен
— Session.php	- сессия
— OAuthProvider.php	- OAuth провайдер
— PasswordReset.php	- запрос на сброс пароля
— Events/	- события
— UserLoggedIn.php	- событие входа
— UserRegistered.php	- событие регистрации
— PasswordReset.php	- событие сброса пароля
— Listeners/	- слушатели событий
— SendWelcomeEmail.php	- отправка welcome email
— LogUserActivity.php	- логирование активности
— NotifyUserLogin.php	- уведомление о входе
— Exceptions/	- обработка исключений
— AuthException.php	- ошибки аутентификации
— InvalidTokenException.php	- невалидный токен
— RateLimitException.php	- превышен лимит запросов
— Providers/	- сервис-провайдеры
— AuthServiceProvider.php	- регистрация auth сервисов
— EventServiceProvider.php	- регистрация событий
— bootstrap/	- загрузчик фреймворка
— app.php	- инициализация приложения
— config/	- конфигурационные файлы
— auth.php	- конфигурация аутентификации
— jwt.php	- настройки JWT (secret, expiration)
— services.php	- внешние сервисы (SMS, OAuth)
— database.php	- подключение к БД
— database/	- миграции и seeders
— migrations/	- миграции БД
— create_users_table.php	- таблица пользователей
— create_tokens_table.php	- таблица токенов
— create_sessions_table.php	- таблица сессий
— create_oauth_providers_table.php	- OAuth провайдеры
— seeders/	- начальные данные
— public/	- публичная директория
— index.php	- точка входа
— routes/	- маршруты API
— web.php	- определение роутов
— secrets/oauth/	- OAuth секреты и ключи (Google, Facebook, VK)
— google.json	- Google OAuth credentials
— facebook.json	- Facebook OAuth credentials
— vk.json	- VK OAuth credentials
— storage/	- хранилище
— logs/	- логи приложения
— lumen.log	- основной лог
— framework/	- кеш фреймворка
— app/	- временные файлы
— tests/	- тесты
— Unit/	- юнит-тесты

└─ Feature/	- функциональные тесты
└─ vendor/	- зависимости Composer

Основные функции:

- Аутентификация: login/logout с JWT токенами;
- Регистрация: создание новых пользователей с валидацией;
- OAuth: интеграция с Google, Facebook, VK для входа через соцсети;
- SMS подтверждение: верификация номера телефона через SMS коды;
- Восстановление пароля: отправка reset ссылок на email;
- Управление сессиями: трекинг активных сессий пользователей;
- JWT токены: генерация access и refresh токенов, автоматическое обновление;
- Rate limiting: защита от brute-force атак;
- События и уведомления: welcome email, логирование активности.

Volumes:

/volumes/auth-dev/secrets/oauth → /var/www/html/secrets/oauth

База данных:

Подключение к MySQL Galera Cluster: 10.88.5.254

База данных: eus_dev_auth

Пользователь: eus_dev_auth

2.2.1.3 Gateway (API Gateway)

Центральная точка входа для всех API-запросов. Выполняет роль API-роутера с использованием Consul для service discovery. Агрегирует запросы к различным микросервисам (auth, api, phone, chat, files-storage, notification и др.).

Хост:

node1

Контейнер:

Имя: deploy-gateway-dev-1

Команда для входа: docker exec -it deploy-gateway-dev-1 /bin/sh

Расположение файлов:

/src/app/	- приложение на Node.js (Koa framework)
└─ src/	- исходный код TypeScript
└─ layers/	- middleware слои (обработчики запросов)
└─ api.ts	- основной API handler
└─ check-auth.ts	- проверка JWT токенов
└─ auth-directly.ts	- прямая аутентификация
└─ cors.ts	- CORS политики
└─ recaptcha.ts	- проверка reCAPTCHA
└─ static.ts	- раздача статики

documentation.ts	- API документация
grafana.ts	- прокси для Grafana
prometheus.ts	- метрики для Prometheus
http-routes/	- определения маршрутов
static/	- статические файлы (аватары, иконки, email шаблоны)
avatars/	- дефолтные аватары пользователей
cards/	- изображения карточек услуг
email/	- HTML шаблоны email
video/	- видео материалы
services-categories/	- иконки категорий услуг
themes/	- темы оформления
build/	- скомпилированный код JavaScript
layers/	- скомпилированные middleware
http-routes/	- скомпилированные маршруты
static/	- копия статических файлов
index.js	- точка входа приложения
node_modules/	- зависимости npm (Koa, Consul, Redis клиенты)
package.json	- описание проекта и зависимостей

Принцип работы Gateway:

- Принимает все входящие API запросы;
- Использует Consul для service discovery (находит адреса микросервисов);
- Маршрутизирует запросы на соответствующие сервисы (auth, api, phone, chat и др.);
- Проверяет JWT токены через middleware check-auth;
- Добавляет CORS заголовки;
- Кеширует результаты через Redis;
- Логирует в Loki, отправляет метрики в Prometheus.

Volumes:

/gateway-dev/documentation → /doc (документация API)

Зависимости:

Redis: deploy-gateway-redis-dev-1 (кеширование)

2.2.1.4 Phone (Телефония)

Сервис интеграции с АТС Asterisk. Обработывает входящие/исходящие звонки, записи разговоров, IVR-меню, статистику звонков. Интегрируется с files-storage для хранения записей.

Хост:

node1

Контейнер:

Имя: deploy-phone-dev-1

Команда для входа: docker exec -it deploy-phone-dev-1 /bin/sh

Расположение файлов:

/src/app/	- приложение на Node.js (TypeScript)
├── src/	- исходный код TypeScript
│ ├── controllers/	- контроллеры (обработчики HTTP запросов)
│ │ └── PhoneController	- обработка запросов звонков
│ ├── services/	- бизнес-логика
│ │ ├── AsteriskService	- взаимодействие с Asterisk AMI
│ │ ├── CallService	- управление звонками
│ │ ├── RecordingService	- работа с записями разговоров
│ │ └── IVRService	- IVR меню
│ ├── repositories/	- работа с БД (TypeORM)
│ │ ├── CallRepository	- история звонков
│ │ ├── QueueRepository	- очереди вызовов
│ │ └── RecordRepository	- записи разговоров
│ ├── entities/	- TypeORM сущности (модели БД)
│ │ ├── Call.ts	- звонок
│ │ ├── CallRecord.ts	- запись разговора
│ │ ├── Queue.ts	- очередь
│ │ └── Extension.ts	- внутренний номер
│ ├── api/	- REST API эндпоинты
│ │ └── routes.ts	- определение маршрутов
│ ├── messages/	- сообщения и события
│ │ └── CallEvent.ts	- события звонков
│ ├── formatters/	- форматирование данных
│ │ └── CallFormatter.ts	- форматирование звонков для API
│ ├── console/	- CLI команды
│ │ └── SyncQueues.ts	- синхронизация очередей с Asterisk
│ ├── tools/	- утилиты
│ │ ├── AmiClient.ts	- клиент Asterisk Manager Interface
│ │ └── Logger.ts	- логирование
│ ├── constants/	- константы (статусы, типы звонков)
│ └── test/	- тесты
├── build/	- скомпилированный код JavaScript
│ ├── controllers/	- скомпилированные контроллеры
│ ├── services/	- скомпилированные сервисы
│ ├── repositories/	- скомпилированные репозитории
│ └── [остальные модули]	- компиляция всех src модулей
├── node_modules/	- зависимости npm
│ └── [asterisk-manager, typeorm, koa и др.]	
└── package.json	- конфигурация проекта и зависимостей

Основные функции:

- AMI (Asterisk Manager Interface) - управление АТС в реальном времени;
- Обработка входящих/исходящих звонков;
- IVR меню и автоматические сценарии;
- Запись разговоров и их хранение;
- Очереди вызовов и распределение звонков;
- Статистика звонков (пропущенные, отвеченные, длительность);
- Интеграция с Files Storage для хранения записей.

Volumes:

- /phone-dev/documentation → /doc
- /nfs → /nfs (readonly, для доступа к записям звонков)

База данных:

Подключение к MySQL Galera Cluster: 10.88.5.254

База данных: eus_dev_phone

Пользователь: eus_dev_phone

Интеграция с Asterisk:

AMI (Asterisk Manager Interface):

- Host: 10.0.2.10
- User: asterisk-manager
- Port: 5038

2.2.1.5 Files Storage (Файловое хранилище)

Централизованное хранилище файлов для всех сервисов. Обрабатывает загрузку, конвертацию изображений, транскрипцию аудио. Интегрируется с converter и transcripтер сервисами.

Хост:

node1

Контейнер:

Имя: deploy-files-storage-dev-1

Команда для входа: docker exec -it deploy-files-storage-dev-1 /bin/sh

Расположение файлов:

```

/src/app/
├── src/
│   ├── file/
│   │   ├── FileService.ts - бизнес-логика файлов
│   │   ├── FileController.ts - API контроллер
│   │   ├── FileRepository.ts - работа с БД
│   │   └── FileUploader.ts - загрузка файлов
│   ├── converter/
│   │   ├── ConverterService.ts - интеграция с converter сервисом
│   │   ├── ImageProcessor.ts - обработка изображений
│   │   └── ThumbnailGenerator.ts - генерация превью
│   ├── transcription/
│   │   ├── TranscriptionService.ts - распознавание речи
│   │   ├── AudioProcessor.ts - обработка аудио
│   │   └── TranscriptionRepository.ts - хранение результатов
│   ├── static-files/
│   │   ├── StaticController.ts - раздача статики
│   │   └── CacheManager.ts - кеширование
│   ├── zip/
│   │   ├── ZipService.ts - создание и распаковка ZIP
│   │   └── ZipController.ts - API для архивов
│   └── common/
│       └── FileValidator.ts - валидация файлов
└── - приложение на Node.js (NestJS framework)
    - исходный код TypeScript
    - управление файлами

```

└─	MimeTypeDetector.ts	- определение типов файлов
└─	StorageManager.ts	- управление хранилищем
└─ dist/		- скомпилированный код
└─ file/		- скомпилированные модули
└─ converter/		- конвертация
└─ transcription/		- транскрипция
└─ [остальные модули]		- компиляция всех src модулей
└─ typings/		- TypeScript определения типов
└─ test/		- тесты
└─ node_modules/		- зависимости npm
└─	[@nestjs, typeorm, sharp, ffmpeg-static и др.]	
└─ package.json		- конфигурация проекта
/uploads/		- хранилище загруженных файлов
└─ images/		- изображения
└─ documents/		- документы (PDF, DOC, DOCX)
└─ audio/		- аудиофайлы
└─ video/		- видеофайлы
└─ temp/		- временные файлы
/nfs/		- монтированное NFS-хранилище (readonly)
└─ asterisk/recordings/		- записи звонков из Asterisk

Основные функции:

- Загрузка и валидация файлов (ограничения по размеру, типам);
- Конвертация изображений (resize, crop, watermark);
- Генерация thumbnails (превью изображений);
- Транскрипция аудио (speech-to-text);
- Создание и распаковка ZIP архивов;
- Хранение метаданных файлов в БД;
- Интеграция с NFS для доступа к записям звонков;
- CDN-like раздача статических файлов с кешированием.

Volumes:

- /volumes/files-storage-dev/uploads → /uploads
- /nfs → /nfs (readonly)

База данных:

Подключение к MySQL Galera Cluster: 10.88.5.254

База данных: eus_dev_files_storage

Пользователь: eus_dev_files_storage

2.2.1.6 Chat (Чат-сервис)

Сервис реального времени для чатов между пользователями и специалистами. Поддерживает WebSocket соединения, хранение истории сообщений, поиск по Manticore, проверку текстов через LanguageTool.

Хост:

node2

Контейнер:

Имя: chat-stack_chat-dev.1.cabuudhcnj18ltpwfe12232a (Docker Swarm service)

Команда для входа: `docker exec -it chat-stack_chat-dev.1.cabuudhcnj18ltpwfe12232a /bin/sh`

Особенности развертывания через Docker Swarm:

Chat сервис развернут как Docker Swarm service (в отличие от других сервисов на Docker Compose), это обеспечивает:

- Автоматическое масштабирование (можно запустить несколько реплик);
- Load balancing между репликами через встроенный Swarm load balancer;
- Автоматический restart при падении контейнера;
- Rolling updates без downtime.

Подключение к Chat сервису:

1. Из других контейнеров на node2: через service name chat-stack_chat-dev (Swarm автоматически балансирует);
2. Из контейнеров на node1: через внешний IP node2 и exposed порт;
3. Список всех реплик: `docker service ps chat-stack_chat-dev` (на node2);
4. Логи всех реплик: `docker service logs chat-stack_chat-dev` (на node2);
5. Масштабирование: `docker service scale chat-stack_chat-dev=3` (запуск 3 реплик).

Расположение файлов:

```
/src/app/
├── src/
│   ├── chat/
│   │   ├── ChatService.ts - бизнес-логика чатов
│   │   ├── ChatController.ts - REST API для чатов
│   │   ├── ChatRepository.ts - работа с БД чатов
│   │   └── ChatRoom.ts - управление комнатами
│   ├── message/
│   │   ├── MessageService.ts - логика сообщений
│   │   ├── MessageController.ts - API сообщений
│   │   ├── MessageRepository.ts - хранение сообщений
│   │   └── MessageFormatter.ts - форматирование
│   ├── user/
│   │   ├── UserService.ts - управление пользователями
│   │   └── UserPresence.ts - онлайн-статус пользователей
│   ├── pinned-message/
│   │   ├── PinnedMessageService.ts - логика закрепления
│   │   └── PinnedMessageRepository.ts - хранение
│   ├── file/
│   │   ├── FileService.ts - загрузка файлов в чат
│   │   └── FileController.ts - API для файлов
│   └── service/
│       ├── ManticoreSearchService.ts - полнотекстовый поиск
│       ├── LanguageToolService.ts - проверка грамматики
│       └── NotificationService.ts - уведомления о новых сообщениях
```

— graphql/	- GraphQL API
— schema.graphql	- схема GraphQL
— resolvers/	- резолверы запросов
— types/	- типы данных
— common/	- общие утилиты
— redis-client.ts	- клиент Redis для pub/sub
— db-connection.ts	- подключение к БД
— logger.ts	- логирование
— migrations/	- миграции БД
— tests/	- тесты
— socket.ts	- Socket.io обработчики (WebSocket)
— routes.ts	- REST API маршруты
— app.ts	- инициализация Koa приложения
— queue.ts	- очереди задач
— index.ts	- точка входа
— node_modules/	- зависимости npm
— [socket.io, koa, typeorm, manticoresearch, ioredis и др.]	
— package.json	- конфигурация проекта

Основной функционал:

- WebSocket соединения через Socket.io (real-time обмен сообщениями);
- REST API для истории чатов и управления;
- GraphQL API для сложных запросов;
- Полнотекстовый поиск по сообщениям через Manticore Search;
- Проверка грамматики через LanguageTool;
- Поддержка файлов в сообщениях (изображения, документы);
- Закрепленные сообщения в чатах;
- Онлайн-статусы пользователей (presence);
- Pub/Sub через Redis для синхронизации между инстансами.

База данных:

Подключение к MySQL Galera Cluster: 10.88.5.254

База данных: eus_dev_chat

Пользователь: eus_dev_chat

Зависимости:

- Redis: chat-stack_chat-redis-dev.1.* (кеширование и pub/sub);
- Manticore: chat-stack_manticore-dev.1.* (поиск по сообщениям);
- LanguageTool: chat-stack_languagetool.1.* (проверка грамматики).

2.2.1.7 CRM (Система управления клиентами)

Веб-интерфейс для сотрудников компании. Управление клиентами, задачами, консультациями, телефонией, аналитикой. React SPA с использованием React-Admin и Material-UI.

Хост:

node1

Контейнер:

Имя: deploy-frontend-monorepo-crm-dev-1

Команда для входа: `docker exec -it deploy-frontend-monorepo-crm-dev-1 /bin/sh`

Расположение файлов:

```
/src/                                     - монорепозиторий frontend-monorepo
├── packages/                             - пакеты монорепозитория
│   └── crm/                             - код CRM приложения (React 16.14 + TypeScript)
│       └── src/                         - исходный код React/TypeScript
│           ├── appRoutesAndResources/ - маршруты и ресурсы приложения
│           │   ├── routes.tsx - конфигурация роутинга
│           │   └── resources.tsx - определение ресурсов React-Admin
│           ├── areas/                  - модуль рабочих зон (территориальные единицы)
│           │   ├── AreaList.tsx - список зон
│           │   ├── AreaEdit.tsx - редактирование зоны
│           │   └── AreaService.ts - бизнес-логика зон
│           ├── auth/                   - аутентификация и авторизация
│           │   ├── LoginPage.tsx - страница входа
│           │   ├── AuthProvider.ts - провайдер авторизации
│           │   └── PermissionChecker.ts - проверка прав доступа
│           ├── clients/                - управление клиентами (основной модуль)
│           │   ├── ClientList.tsx - список клиентов
│           │   ├── ClientShow.tsx - просмотр клиента
│           │   ├── ClientEdit.tsx - редактирование клиента
│           │   ├── cards/              - карточки клиентов
│           │   │   ├── CardList.tsx - список карточек
│           │   │   ├── CardTimeline.tsx - таймлайн событий карточки
│           │   │   └── CardService.ts - API карточек
│           │   ├── cases/              - кейсы/дела клиентов
│           │   │   ├── CaseList.tsx - список дел
│           │   │   ├── CaseShow.tsx - просмотр дела
│           │   │   └── CaseService.ts - логика дел
│           │   ├── customerRetention/ - удержание клиентов
│           │   │   ├── RetentionDashboard.tsx - дашборд удержания
│           │   │   └── RetentionService.ts - логика удержания
│           │   └── ClientService.ts - API клиентов
│           ├── tasks/                  - задачи
│           │   ├── TaskList.tsx - список задач
│           │   ├── TaskBoard.tsx - канбан-доска задач
│           │   ├── TaskCalendar.tsx - календарь задач
│           │   ├── TaskEdit.tsx - редактирование задачи
│           │   └── TaskService.ts - API задач
│           ├── chat/                   - модуль чатов
│           │   ├── ChatWindow.tsx - окно чата
│           │   ├── ChatList.tsx - список чатов
│           │   ├── MessageInput.tsx - ввод сообщений
│           │   └── ChatService.ts - Socket.io клиент
│           ├── video/                  - видеозвонки
│           │   ├── VideoCall.tsx - интерфейс видеозвонка
│           │   ├── WebRTCService.ts - WebRTC логика
│           │   └── CallControls.tsx - управление звонком
│           └── statistics/              - статистика и аналитика
│               └── call/                - статистика звонков
│                   ├── CallStatsDashboard.tsx
│                   └── CallStatsService.ts
```

- clients/ - статистика клиентов
 - ClientStatsDashboard.tsx
 - ClientGrowthChart.tsx
- tasks/ - статистика задач
 - TaskStatsDashboard.tsx
 - TaskCompletionChart.tsx
- users/ - статистика сотрудников
 - UserPerformance.tsx
 - UserActivityChart.tsx
- documents/ - документы
 - DocumentList.tsx - список документов
 - DocumentViewer.tsx - просмотр документов
 - DocumentService.ts - API документов
- legalCloud/ - юридическое облако (документооборот)
 - LegalCloudBrowser.tsx - файловый браузер
 - DocumentUpload.tsx - загрузка документов
 - LegalCloudService.ts - API облака
- sales/ - продажи и коммерция
 - SalesDashboard.tsx - дашборд продаж
 - SalesService.ts - API продаж
 - PaymentManagement.tsx - управление платежами
- settings/ - настройки CRM
 - GeneralSettings.tsx - общие настройки
 - UserSettings.tsx - настройки пользователей
 - IntegrationSettings.tsx - настройки интеграций
- users/ - управление пользователями
 - UserList.tsx - список пользователей
 - UserEdit.tsx - редактирование пользователя
 - UserService.ts - API пользователей
- tables/ - переиспользуемые таблицы
 - DataGrid.tsx - универсальная таблица
 - FilterPanel.tsx - панель фильтров
 - ExportButton.tsx - экспорт данных
- components/ - переиспользуемые компоненты
 - Layout/ - компоненты layout
 - Forms/ - формы и inputs
 - Modals/ - модальные окна
 - Common/ - общие компоненты
- api/ - API клиент
 - client.ts - HTTP клиент (axios/fetch)
 - endpoints.ts - определения эндпоинтов
 - interceptors.ts - перехватчики запросов
- store/ - состояние приложения (Redux/MobX)
 - slices/ - Redux slices
 - actions/ - Redux actions
- utils/ - утилиты
 - date.ts - работа с датами
 - format.ts - форматирование
 - validation.ts - валидация
- types/ - TypeScript типы
 - Client.ts - типы клиентов
 - Task.ts - типы задач
 - [остальные типы]
- App.tsx - корневой компонент
- index.tsx - точка входа React
- dist/ - собранные файлы (production build)
 - index.html - главный HTML
 - assets/ - JS, CSS, изображения
 - manifest.json - манифест приложения
- assets/ - статические ресурсы (иконки, аудио, badges)

icons/	- иконки
audio/	- звуки уведомлений
badges/	- бейджи статусов
public/	- публичные файлы (копируются в dist)
node_modules/	- зависимости пакета
package.json	- конфигурация CRM пакета
tsconfig.json	- конфигурация TypeScript
vite.config.ts	- конфигурация Vite
...	- другие пакеты монорепозитория
node_modules/	- общие зависимости монорепозитория
package.json	- описание монорепозитория
yarn.lock	- lockfile зависимостей
turbo.json	- конфигурация Turborepo (опционально)

Сборщик:

Vite

Веб-сервер в контейнере:

Serve (Node.js статический сервер)

Точка входа:

/src/packages/crm/dist/index.html

2.2.1.8 Личный кабинет

Веб-интерфейс для пользователей платформы. Доступ к своим данным, консультациям, чату с консультантом, просмотр статуса услуг.

Хост:

node2

Контейнер:

Имя: deploy-lk-dev-1

Команда для входа: `docker exec -it deploy-lk-dev-1 /bin/sh`

Расположение файлов:

/src/	- монорепозиторий
packages/	- пакеты монорепозитория
lk/	- код ЛК приложения
src/	- исходный код React/TypeScript
dist/	- собранные файлы
assets/	- статические ресурсы
node_modules/	- зависимости
...	- другие пакеты
node_modules/	- общие зависимости
package.json	- описание монорепозитория
yarn.lock	- lockfile

Сборщик:

Webpack

Веб-сервер:

Nginx (встроен в контейнер)

2.2.2 Инфраструктурные сервисы

1. Nginx-proxy. Reverse proxy, распределяет запросы по доменам на различные контейнеры

Хост: node1

Контейнер: deploy_nginx-proxy_1

Конфигурация: /etc/nginx/vhost.d (из ./locations)

2. Consul. Service discovery для динамической маршрутизации в Gateway

Хост: node1

Контейнер: deploy-consul-1

3. Event-bus. Redis для pub/sub между сервисами

Хост: node1

Контейнер: deploy-event-bus-dev-1

4. Manticore. Поисковый движок для задач и чатов

Хост: node1

Контейнер: deploy-manticore-dev-1

Backup: /backups/manticore-dev

5. Converter. Конвертация изображений для files-storage

Хост: node1

Контейнер: deploy_converter_1

2.2.3 Мониторинг и наблюдаемость

1. Loki. Агрегация логов

Хост: node1, node2

Контейнер: deploy_loki_1 (node1), chat-stack_loki.1.* (node2)

Порт: 3100

Конфигурация: /etc/loki/real-config.yaml

2. Tempo. Distributed tracing

Хост: node1

Контейнер: deploy_tempo_1

Порты: 3200 (HTTP), 4317 (OTLP gRPC)

3. Prometheus. Сбор метрик

Хост: node1, node2

Контейнер: deploy_prometheus_1 (node1), chat-stack_prometheus.1.* (node2)

Порт: 9090

4. Exporters

- Node Exporter: `deploy_nodeexporter_1` (node1), `chat-stack_nodeexporter.1.*` (node2) - метрики хоста;
- Nginx Exporter: `nginx-exporter` (node1) - метрики nginx;
- Docker Exporter: `docker_exporter` (node1), `chat-stack_docker_exporter.1.*` (node2) - метрики контейнеров;
- MySQL Exporter: `mysql-exporter` (node1) - метрики БД;
- cAdvisor: `deploy-cadvisor-1` (node1), `chat-stack_cadvisor.1.*` (node2) - метрики контейнеров.

5. Pyroscope. Continuous profiling

Хост: node1

Контейнер: `deploy_pyroscope_1`

Порт: 4040

2.2.4 Команды для быстрого доступа к контейнерам

Общие команды после подключения по SSH:

```
# Просмотр всех контейнеров
docker ps

# Просмотр логов контейнера
docker logs <container_name>

# Вход в контейнер
docker exec -it <container_name> /bin/sh
```

Примеры для node1:

```
# Вход в API контейнер
docker exec -it deploy-api-dev-1 /bin/sh

# Вход в Gateway
docker exec -it deploy-gateway-dev-1 /bin/sh

# Вход в CRM
docker exec -it deploy-frontend-monorepo-crm-dev-1 /bin/sh
```

Примеры для node2:

```
# Вход в ЛК
docker exec -it deploy-lk-dev-1 /bin/sh

# Вход в Chat (имя может отличаться, используйте docker ps)
```

```
docker exec -it chat-stack_chat-dev.1.cabuudhcnjjl8ltpwfe12232a /bin/sh
```

2.2.5 Полезные команды внутри контейнеров

API (Laravel):

```
# Просмотр логов
tail -f /var/www/html/storage/logs/laravel-dev.log

# Artisan команды
php artisan route:list
php artisan config:cache
```

Frontend (CRM, ЛК):

```
# Просмотр собранных файлов
ls -la /src/packages/crm/dist
ls -la /src/packages/lk/dist
```

Базы данных:

```
# Подключение к MySQL из контейнера БД
mysql -uroot -psecret <database_name>

# Подключение к MySQL из хоста
docker exec -it <db_container_name> mysql -uroot -psecret <database_name>
```

3 КОНТАКТЫ ТЕХНИЧЕСКИХ СПЕЦИАЛИСТОВ

Контактные данные технического специалиста, уполномоченного консультировать относительно процедуры установки, конфигурации и эксплуатации программного продукта:

Песковацков Денис Владимирович, веб-разработчик	Эл. почта: d.peskovackov@els24.com Telegram: @deiv_1980
---	---