

ИНСТРУКЦИЯ ПО УСТАНОВКЕ И ЭКСПЛУАТАЦИИ

СОЗ «ТАЙГА»

Версия документа: 1.2 от 10.09.2026

Код: HMA-001-INSTRUCT

г. Москва, 2026 г.

СОДЕРЖАНИЕ

1. Термины и сокращения.....	5
2. Краткое описание ПО и его назначение.....	7
2.1. Основные функции.....	7
2.1.1. Создание записок.....	7
2.1.2. Доступ к запискам.....	7
2.1.3. Редактирование и управление.....	7
2.1.4. Особенности.....	7
2.1.5. Назначение.....	8
2.1.6. Целевая аудитория документа.....	8
3. Системные требования.....	9
3.1. Операционная система.....	9
3.2. Зависимости.....	9
3.2.1. База данных.....	9
3.2.2. Дополнительные инструменты.....	10
3.2.3. Оборудование.....	10
4. Установка и настройка.....	12
4.1. Настройка PostgreSQL.....	12
4.1.1. Рекомендации по размещению.....	12
4.1.2. Установка PostgreSQL в Linux.....	12
Ubuntu / Debian / Astra Linux.....	12
Fedora / Ред ОС.....	12
4.1.3. Создание базы данных и пользователя.....	12
4.1.4. Проверка подключения.....	13
4.1.5. Настройка pg_hba.conf (аутентификация).....	13
Ubuntu / Debian / Astra Linux.....	13
Fedora / Ред ОС.....	13
Редактирование pg_hba.conf.....	14
4.2. Конфигурационные файлы.....	15
4.2.1. Создание файла конфигурации.....	15
4.2.2. Базовая структура config.json.....	15
4.2.3. Описание параметров конфигурации.....	16
db — параметры подключения к PostgreSQL.....	16

http — настройки HTTP-сервера	17
captcha — настройки CAPTCHA	18
logging — настройки логирования	18
jwt — настройки JWT-токенов	19
protected — хранилище мастер-ключей (обязательная секция)	20
cache — параметры Redis (опциональная секция)	21
4.2.4. Переменные окружения	23
4.2.5. Проверка конфигурации	23
4.3. Первичная инициализация хранилища ключей ()	24
4.4. Запуск сервера через бинарный файл	25
4.5. Запуск через системные службы (systemd)	26
4.5.1. Для Ubuntu / Debian / Astra Linux	27
1. Подготовка	27
2. Создание файла службы	27
3. Активация и запуск службы	28
4. Проверка состояния	28
5. Управление службой	29
5. Запуск и использование	29
5.1. Первоначальная настройка	29
5.2. Демонстрационные сценарии	33
5.2.1. Тестовый сценарий: Создание и чтение записки	33
Откройте главную страницу	34
Введите содержимое записки	34
Настройте записку	35
Создайте записку	36
Дополнительные действия	37
6. Специфические особенности	38
6.1. Миграции базы данных	38
6.2. Шифрование и ключи	38
6.3. Разграничение доступа	38
6.4. Лицензирование	39
7. Системное логирование	40
7.1. Уровни логирования	40
8. Устранение неполадок: типичные ошибки и их решение	41

8.1. Ошибки при запуске сервера	41
8.1.1. Ошибка: «Port is already in use» (порт занят)	41
8.2. Ошибки при подключении к PostgreSQL	41
8.2.1. Ошибка: «could not connect to server» (не подключается к БД)	41
8.3. Ошибки инициализации хранилища ключей и конфигурации	42
8.3.1. Ошибка: «protected.key_dir is required» (не задан каталог ключей)	42
8.3.2. Ошибка: «key repository directory does not exist: ... — run --setup-keys first» (хранилище ключей не инициализировано)	42
8.3.3. Ошибка: «jwt secret is required» / «jwt secret must be at least 32 characters»	43
8.4. Ошибки в браузере	43
8.4.1. Ошибка: "WASM modules not loaded" (не загружены WASM модули)	43
8.5. Ошибки при работе с CAPTCHA	43
8.5.1. Ошибка: "captcha validation failed" (CAPTCHA не проходит)	43
8.6. Ошибки при работе с LDAP	44
8.6.1. Ошибка: «LDAP connection failed» (не подключается к LDAP)	44
8.7. Ошибки при работе с кэшем (Redis)	44
8.7.1. Ошибка: "cache initialization failed" (кэш не инициализирован)	44
8.8. Ошибки в логировании	45
8.8.1. Ошибка: «logs not output» (логи не выводятся)	45
9. Поддержка	47
9.1. Контакты	47

1. ТЕРМИНЫ И СОКРАЩЕНИЯ

СОЗ «ТАЙГА» — Сервис одноразовых записок «Тайга», защищённое решение для одноразовой передачи конфиденциальных сообщений с использованием криптографических методов.

Оконечное (сквозное) шифрование (end-to-end encryption, E2EE) — окончное шифрование; метод защиты данных, при котором сообщения шифруются на устройстве отправителя и расшифровываются только на устройстве получателя. Все операции происходят в браузере без передачи незашифрованных данных на сервер.

ГОСТ 34.10-2018 — отечественный стандарт, определяющий алгоритмы формирования и проверки электронной цифровой подписи (ЭЦП).

ГОСТ 34.11-2018 — отечественный стандарт хэширования данных, также известный как «Стрибог».

ГОСТ 34.12-2018 — отечественный стандарт для симметричного блочного алгоритма шифрования «Кузнечик».

ЭЦП (Электронная цифровая подпись) — механизм подписания данных с использованием криптографических ключей, обеспечивающий аутентичность и неотказуемость информации.

Записка — одноразовая зашифрованная информация, созданная через сервис Тайга. Может содержать текст, файлы и быть защищена различными методами шифрования.

Ссылка на записку — уникальный URL с ключом расшифрования, позволяющий получить доступ к конкретной записке. После прочтения записка автоматически удаляется.

Ключ расшифрования — секретный элемент, необходимый для декодирования и расшифрования зашифрованного контента. Может быть дополнительным образом защищен паролем или сертификатом.

API (Application Programming Interface) — интерфейс прикладного программирования; набор определений и протоколов для построения и интеграции программных приложений.

AES (Advanced Encryption Standard) — симметричный алгоритм шифрования данных, принятый в качестве стандарта правительством США.

CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) — тест, предназначенный для проверки того, что пользователь является человеком, а не ботом.

UI (User Interface) — пользовательский интерфейс; совокупность компонентов системы, взаимодействующих с человеком через браузерный веб-интерфейс.

Markdown — облегчённая разметка текста, позволяющая форматировать содержимое записки (заголовки, списки, выделение текста и т.д.).

PoW (Proof of Work) — доказательство через решение небольшой задачи.

WASM (WebAssembly) — низкоуровневый формат кода, используемый для ускорения работы в браузере через специализированные модули.

Rutoken — сертифицированное криптографическое средство защиты информации, позволяющее использовать аппаратные ключи для расшифровки записок.

FIDO2 — стандарт аутентификации без паролей через криптографические ключи, доступный в системе Тайга для дополнительной защиты ссылок.

LDAP/AD — протоколы аутентификации и авторизации, интегрируемые с системой Тайга для корпоративного использования.

POSTGRESQL — объектно-реляционная система управления базами данных, используемая в Тайга для хранения зашифрованных записок и пользовательских данных.

REDIS — нереляционная (NoSQL) система управления базами данных с оперативной памятью, применяемая как кэш для ускорения запросов и хранения CAPTCHA.

Опциональный компонент — приложение может работать без Redis, но с сниженной производительностью.

XOR (Exclusive OR) — логическая операция, используемая для побитового шифрования данных. Также применяется как дополнительный способ защиты ссылок на записки.

2. КРАТКОЕ ОПИСАНИЕ ПО И ЕГО НАЗНАЧЕНИЕ

Сервис одноразовых записок «Тайга» (СОЗ «Тайга») — это веб-приложение для безопасной передачи одноразовых зашифрованных записок с использованием *оконечного шифрования*. Система обеспечивает конфиденциальность данных через интерактивный (браузерный) интерфейс на стороне клиента. Все операции по созданию, передаче и чтению записок происходят без отправки незашифрованных данных на сервер.

2.1. Основные функции

2.1.1. Создание записок

- Пользовательский интерфейс позволяет вводить текст, прикреплять файлы и использовать Markdown разметку.
- Выбор метода шифрования: XOR, AES, ГОСТ 34.12-2018 (Кузнечик).
- Настройка времени жизни записки (автоматическое удаление после определённого периода).

2.1.2. Доступ к запискам

- Получение уникальной ссылки с ключом для расшифрования зашифрованного контента.
- Дополнительная защита ключа в ссылке через пароль, сертификаты (Rutoken, WebAuthn/FIDO2 аутентификатор) или мнемонические фразы.

2.1.3. Редактирование и управление

- Возможность редактирования записки при наличии специального ключа.
- Управление сертификатами: импорт/экспорт, хранение в локальном хранилище (localStorage), доверенные сертификаты

2.1.4. Особенности

- **Оконечное шифрование:** Все данные шифруются в браузере перед отправкой, обеспечивая максимальную безопасность. Сервер является хранилищем шифротекстов.
- **Одноразовое использование:** записки могут быть прочитаны только один раз и автоматически удаляются после этого.
- **Многоязычный интерфейс:** Поддержка русского и английского языков для удобства пользователей.

2.1.5. Назначение

Приложение предназначено для безопасного обмена информацией между пользователями, где важно обеспечить защиту данных от несанкционированного доступа. Может использоваться:

- В личных целях: хранение и передача личных записок.
- В корпоративной среде: безопасная коммуникация по юридическим или техническим документам, передача чувствительных данных: пароли, токены, базы данных.

2.1.6. Целевая аудитория документа

- **Конечные пользователи:** для создания и получения зашифрованных сообщений.
- **Администраторы/разработчики:** для настройки системы, интеграции с LDAP/AD и управления провайдерами аутентификации.

3. СИСТЕМНЫЕ ТРЕБОВАНИЯ

3.1. Операционная система

- **Linux** (Ubuntu 20.04 LTS или Debian 11):
 - Предпочтительная платформа для развертывания и тестирования.
 - Архитектура: x86_64, arm, arm64, 386.
- **Astra Linux** (1.6+):
 - Предпочтительная отечественная платформа для развертывания.
 - Архитектура: x86_64, arm, arm64, 386.
- **Red OS** (3.0+):
 - Архитектура: x86_64, arm, arm64, 386.
- **Windows Server**:
 - Поддержка возможна, но требует дополнительной конфигурации и проверки совместимости.

Примечание:

Совместимость СОЗ «Тайга» с отечественными операционными системами Astra Linux (1.6+) и Red ОС (3.0+) обеспечивается и регулярно проверяется правообладателем при выпуске поставочных сборок и приёмочных испытаниях.

Подтверждающие документы по результатам специальных испытаний совместимости с операционными системами, соответствующими требованиям к доверенному программному обеспечению, будут представлены правообладателем дополнительно по мере их оформления.

Перечни поддерживаемых архитектур процессора (x86_64, arm, arm64, 386) и минимальная версия PostgreSQL (≥ 13) задаются правообладателем как матрица поддержки; готовые сборки поставляются правообладателем.

3.2. Зависимости

3.2.1. База данных

- **PostgreSQL ≥ 13 :**
 - Основное хранилище данных (записки, пользователи, сертификаты).

- Отдельная схема не требуется: сервис работает в схеме `public`, таблицы создаются встроенными миграциями автоматически при запуске.
- Пользователю БД нужны права на создание таблиц и последовательностей в схеме `public` (триггеры сервисом не используются).
- Проверена работа с PostgreSQL 17.

3.2.2. Дополнительные инструменты

- Расширение для браузера **Рутокен-плагин**:
 - Только для расшифрования с помощью токенов Рутокен.
- **Redis** (опционально):
 - Рекомендуется для продуктовой среды для ускорения работы и хранения САРТЧНА.
 - Без Redis приложение работает, но с сниженной производительностью (используется встроенный in-memory кэш silverbox).
 - **Важно:** silverbox используется только при **отсутствии** секции `cache` в `config.json`. Если секция `cache` задана, Redis обязателен — при его недоступности сервис не запустится.

3.2.3. Оборудование

- **Процессор**:
 - Минимум 2 ядра (x86_64, arm, arm64, 386 архитектура).
 - Рекомендуется 4+ для нагрузок и параллельных операций.
- **Память ОЗУ**:
 - Минимум 4 ГБ (для тестирования).
 - Рекомендуется 8 ГБ для продакшна и высокой нагрузки.
- **Дисковое пространство**:
 - Минимум 10 ГБ свободного места.
 - Для установки и логов.
- **База данных**:
 - Минимум 10 ГБ свободного места.
 - Рекомендуемый рассчитывается индивидуально, в зависимости от нужд и количества пользователей.

- **Сеть:**
 - Поддержка TCP/IP для взаимодействия с PostgreSQL, LDAP серверами и API.

Примечание:

Для полнофункциональной работы требуется:

- 1. Доступ к PostgreSQL DB (по умолчанию на localhost)*
- 2. Поддержка WebAssembly в браузере пользователей для ускорения шифрования в браузере*

4. УСТАНОВКА И НАСТРОЙКА

4.1. Настройка PostgreSQL

4.1.1. Рекомендации по размещению

Рекомендация:

Для продуктовой среды рекомендуется устанавливать PostgreSQL на **отдельном сервере**. Это обеспечивает:

- Более высокую производительность за счёт разделения вычислительных ресурсов
- Улучшенную безопасность и изоляцию базы данных
- Возможность независимого масштабирования
- Упрощённое резервное копирование и восстановление

4.1.2. Установка PostgreSQL в Linux

Ubuntu / Debian / Astra Linux

```
sudo apt update -y
sudo apt install postgresql postgresql-contrib -y
sudo systemctl start postgresql
sudo systemctl enable postgresql
```

Fedora / Ред ОС

```
sudo dnf install postgresql-server postgresql-contrib -y
sudo postgresql-setup --initdb
sudo systemctl start postgresql
sudo systemctl enable postgresql
```

Примечание:

В некоторых версиях Ред ОС может использоваться `yum` вместо `dnf`.

4.1.3. Создание базы данных и пользователя

Подключитесь к PostgreSQL:

```
sudo -i -u postgres
psql
```

Создайте пользователя и БД

```
CREATE USER taiga_user WITH PASSWORD 'secure_password';
CREATE DATABASE taiga_db OWNER taiga_user;
```

Примечание:

Замените *secure_password* на надёжный пароль перед использованием в продуктовой среде.

Назначьте права

```
-- Подключитесь к базе данных
\c taiga_db

-- Дайте права только на необходимые схемы и объекты
GRANT USAGE ON SCHEMA public TO taiga_user;
GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA public TO taiga_user;
GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA public TO taiga_user;

-- Для будущих объектов настройте дефолтные привилегии
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT SELECT, INSERT, UPDATE, DELETE ON
TABLES TO taiga_user;
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT USAGE, SELECT ON SEQUENCES TO
taiga_user;

exit;
```

4.1.4. Проверка подключения

Тестовое соединение

```
psql -h localhost -p 5432 -U taiga_user -d taiga_db
```

Примечание:

Убедитесь, что PostgreSQL сервер запущен и доступен на указанном хосте/порте.

Если вы получили ошибку вида:

```
psql: error: connection to server at "localhost" (:::1), port 5432 failed: FATAL:
Ident authentication failed for user "taiga_user"
```

необходимо провести настройку pg_hba.conf

4.1.5. Настройка pg_hba.conf (аутентификация)

Отредактируйте файл pg_hba.conf изменив METHOD с ident на scram-sha-256

Ubuntu / Debian / Astra Linux

```
nano /etc/postgresql/*/main/pg_hba.conf
```

Fedora / Ред ОС

```
nano /var/lib/pgsql/data/pg_hba.conf
```

Редактирование pg_hba.conf

#	TYPE	DATABASE	USER	ADDRESS	METHOD	
# Локальное подключение через Unix socket						
local	all		taiga_user		scram-sha-256	
# Подключение из localhost (рекомендуется для продакшена)						
host	taiga_db		taiga_user	127.0.0.1/32	scram-sha-256	
host	taiga_db		taiga_user	:::1/128	scram-sha-256	
# Если БД на отдельном сервере – разрешите доступ только с конкретного IP приложения						
# host	taiga_db		taiga_user	<IP_ПРИЛОЖЕНИЯ>/32	scram-sha-256	
# Отключите доверенную аутентификацию для удалённых подключений						
# local	all		all		trust	# УДАЛИТЬ
ИЛИ ЗАКОММЕНТИРОВАТЬ						
# host	all		all	0.0.0.0/0	trust	#
УДАЛИТЬ ИЛИ ЗАКОММЕНТИРОВАТЬ						

После изменений перезапустите PostgreSQL:

```
sudo systemctl restart postgresql
```

и проверьте соединение

```
psql -h localhost -p 5432 -U taiga_user -d taiga_db
```

4.2. Конфигурационные файлы

4.2.1. Создание файла конфигурации

Для работы сервиса «ТАЙГА» необходимо создать файл конфигурации `config.json` и при необходимости задать переменные окружения. Конфигурация определяет параметры подключения к базе данных, HTTP-серверу, кэшированию, CAPTCHA, аутентификации (JWT), шифрования чувствительных данных (`protected` — хранилище мастер-ключей) и другим критически важным настройкам.

Секция `protected` **обязательна**: без неё сервис не запускается (см. описание ниже и раздел «Первичная инициализация хранилища ключей»).

Файл `config.json` — это JSON-документ, структура которого полностью соответствует полям объекта конфигурации сервиса.

4.2.2. Базовая структура `config.json`

```
{
  "db": {
    "host": "localhost",
    "port": "5432",
    "user": "taiga_user",
    "pwd": "secure_password",
    "database": "taiga_db",
    "sslmode": "disable"
  },
  "http": {
    "port": ":80",
    "tls": {
      "cert": "cert.pem",
      "key": "key.pem"
    }
  },
  "protected": {
    "key_dir": "/etc/taiga/keys",
    "reset_cert": false
  },
  "captcha": {
    "start_complexity": 6,
    "step_complexity": 3
  },
  "logging": {
    "level": "debug",
    "encoding": "json",
    "output_paths": [
      "stdout"
    ]
  },
}
```

```

"jwt": {
  "secret": "change-this-secret-in-production-to-long-random-string-at-least-32-
chars",
  "access_token_ttl": "15m",
  "refresh_token_ttl": "168h",
  "bcrypt_cost": 13
}
}

```

Примечание:

Секция *cache* (Redis) в базовом примере отсутствует намеренно — она **опциональна**: без неё сервис использует встроенный *in-memory* кэш (*silverbox*). Для продуктовой среды добавьте секцию *cache* (см. таблицу параметров ниже). Секция *options* в конфигурации не используется.

4.2.3. Описание параметров конфигурации

db — параметры подключения к PostgreSQL

Таблица 1

Параметр	Тип	Обязательный	Описание
host	string	да	Адрес сервера PostgreSQL (например, localhost или 192.168.0.1)
port	string	да	Порт PostgreSQL (по умолчанию 5432). Указывается как строка!
user	string	да	Имя пользователя базы данных
pwd	string	да	Пароль пользователя
database	string	да	Имя базы данных (объекты создаются миграциями в схеме public, отдельная

			схема не требуется)
sslmode	string	нет	Режим TLS-подключения к PostgreSQL (например, disable, require). По умолчанию disable

http — настройки HTTP-сервера

Таблица 2

Параметр	Тип	Обязательный	Описание
port	string	да	Порт HTTP-сервера в виде строки с двоеточием (например, ":80", ":8080", ":443"). Число вместо строки — ошибка конфигурации
tls.cert	string	условно	Путь к SSL-сертификату (.pem). Обязателен при использовании HTTPS (порт :443)
tls.key	string	условно	Путь к приватному ключу (.pem). Обязателен при использовании HTTPS

Примечание:

Для HTTPS-режима обязательно укажите tls.cert и tls.key. Порт обычно :443.

captcha — настройки CAPTCHA

Таблица 3

Параметр	Тип	Обязательный	Описание
start_complexity	int	нет	Начальная сложность CAPTCHA (количество проверяемых хешей). По умолчанию 6
step_complexity	int	нет	Шаг увеличения сложности при повторных неудачных попытках. По умолчанию 3

Зачем это нужно? CAPTCHA защищает от автоматических запросов (ботов).

Сложность рассчитывается по формуле $start_complexity + N / step_complexity$ (целочисленное деление), где N — число не протухших CAPTCHA-запросов с данного IP (хранятся в кэше 1 час). При значениях по умолчанию это означает +1 к сложности за каждые 3 запроса с того же IP — частые обращения с одного адреса увеличивают стоимость решения PoW. Это усложняет атаки перебора.

logging — настройки логирования

Таблица 4

Параметр	Тип	Обязательный	Описание
level	string	да	Уровень логирования: trace, debug, info, warn, error, fatal
encoding	string	да	Формат вывода: console

			(человекочитаемый), json (для систем мониторинга)
output_paths	array of string	да	Куда писать логи: ["stdout"], ["/var/log/taiga.log"] или комбинированный список

Пример. Для локального теста используйте console + stdout, для продуктовой среды — json + файлы логов.

jwt — настройки JWT-токенов

Таблица 5

Параметр	Тип	Обязательный	Описание
secret	string	нет	Секретный ключ для подписи JWT. Должен быть не короче 32 символов! Используйте криптографически стойкий случайный генератор (openssl rand -base64 32).
access_token_ttl	string	нет	Время жизни access-токена (например, "15m", "2h", "1d"). По умолчанию 15m
refresh_token_ttl	string	нет	Время жизни refresh-токена (например, "7d", "30d"). По

			умолчанию 168h (7 дней)
bcrypt_cost	int	нет	Стоимость алгоритма bcrypt для хеширования паролей. По умолчанию 13. Значение вне допустимого диапазона (4–31) или отсутствующее поле автоматически заменяются на 13

***Совет.** Чем выше bcrypt_cost, тем медленнее проверка паролей, но тем устойчивее система к брутфорсу. Для продуктовой среды рекомендуется 13–14.*

protected — хранилище мастер-ключей (обязательная секция)

Сервис хранит в БД в зашифрованном виде (ГОСТ мастер-ключом):

- Информацию о записке, хранящую доверенные сертификаты
. Ключи лежат в каталоге, указанном в protected.key_dir.

Таблица 6

Параметр	Тип	Обязательный	Описание
key_dir	string	да	Путь к каталогу хранилища ключей. Относительный путь разрешается от каталога файла конфигурации (не от текущего каталога процесса). Без key_dir сервис не запустится:

			ошибка protected.key_dir is required
reset_cert	bool	нет	Разрешает сброс зашифрованной конфигурации доверенных сертификатов (protected_certs_config) из административной панели. По умолчанию false

Каталог ключей содержит файлы без расширения (0, 1, ...): 0 — основной ключ шифрования, остальные используются для расшифрования данных, зашифрованных предыдущими ключами после ротации.

Важно:

- Секция *protected* обязательна (изменение введено версией с мастер-ключом): старые *config.json* без неё **не поднимутся**.
- Каталог ключей **обязательно включите в схему резервного копирования**: при утрате файла 0 зашифрованные ранее данные восстановить невозможно. Шифруются чувствительные данные, но не сами шифротексты, поэтому утрата не ведёт к потере доступа к запискам.
- Файлы ключей никогда не должны попадать в системы контроля версий и общедоступные хранилища.
- До первого запуска каталог должен быть инициализирован флагом *--setup-keys* (см. раздел «Первичная инициализация хранилища ключей»). Иначе при запуске: *key repository directory does not exist: ... – run --setup-keys first*.

cache — параметры Redis (опциональная секция)

Таблица 7

Параметр	Тип	Обязательный	Описание
host	string	нет	Адрес Redis. По умолчанию

			localhost
port	string	нет	Порт Redis в виде строки . По умолчанию "6379"
user	string	нет	Имя пользователя ACL Redis (если требуется)
pwd	string	нет	Пароль Redis
database	int	нет	Номер БД Redis в виде числа (например, 0, а не "0")
sslmode	string	нет	TLS-режим: disable (по умолчанию), require (без проверки сертификата), verify-ca / verify-full (с проверкой)
ca_file	string	нет	Путь к PEM-сертификату СА для режимов verify-ca / verify-full

Важно. Silverbox (встроенный in-memory кэш) используется **только при отсутствии** секции cache целиком. Если секция cache задана, Redis становится обязательным: при недоступности сервера запуск завершится ошибкой `init redis cache` (вложенная `connect to redis`).

Примечание про options:

Секция `options` в конфигурации пока не используется. Начальный экран приложения определяется **не конфигурацией**, а состоянием инициализации системы: пока

администратор не создан, все запросы перенаправляются на мастер первоначальной настройки `/setup`.

4.2.4. Переменные окружения

Любое значение конфигурации можно переопределить переменной окружения, **имя которой равно пути ключа в конфиге** (через точку):

Таблица 8

Переменная окружения	Переопределяет
<code>db.host</code>	<code>db.host</code>
<code>db.pwd</code>	<code>db.pwd</code>
<code>http.port</code>	<code>http.port</code>
<code>jwt.secret</code>	<code>jwt.secret</code>
<code>protected.key_dir</code>	<code>protected.key_dir</code>

Специального префикса (`TAIGA_`, `TAIGA_HTTP_PORT` и т.п.) **не существует** — такие переменные сервисом не читаются. Значения передаются строками; длительности — в формате `Go` (`15m`, `168h`). В `systemd`-юните задаются, например, так: `Environment=http.port=:8080`.

4.2.5. Проверка конфигурации

Перед запуском убедитесь, что:

1. База данных `taiga_db` создана, пользователь имеет права на создание таблиц и последовательностей в схеме `public` (миграции раскатываются автоматически при запуске).
2. Задана секция `protected` с `key_dir`, и хранилище ключей уже инициализировано запуском с `--setup-keys` (см. следующий раздел).
3. В `jwt.secret` не меньше 32 символов.
4. Для HTTPS-режима (порт `:443`) указаны `tls.cert` и `tls.key`, и файлы существуют.
5. Порт HTTP-сервера не занят другими приложениями.

Примечание:

Ошибки в конфигурации приведут к завершению запуска с сообщением `could not connect to server` (неверные параметры БД), `jwt secret is required` / `jwt secret must be at least 32`

characters (секрет короче 32 символов), protected.key_dir is required (нет пути к хранилищу ключей), key repository directory does not exist: ... — run --setup-keys first (хранилище ключей не инициализировано) или listen tcp: address already in use (порт занят) в логах сервиса. Используйте уровень debug в logging.level, чтобы увидеть подробную информацию.

Примечание:

Настройки LDAP-аутентификации, SMTP-уведомлений и глобальные параметры системы (разрешение анонимных записок, саморегистрация и т.д.) не задаются в config.json. Они настраиваются через административный интерфейс после первого запуска сервиса.

4.3. Первичная инициализация хранилища ключей ()

Перед первым запуском сервиса необходимо один раз инициализировать хранилище мастер-ключей (см. секцию protected выше):

```
./taiga_porcini --config=config.json --setup-keys
```

Команда создаёт каталог protected.key_dir и генерирует в нём начальные ключи (0 — основной ключ шифрования, остальные — для расшифрования после ротации). Без этого шага сервер не запустится с ошибкой key repository directory does not exist: ... — run --setup-keys first.

Сервис поддерживает и другие служебные режимы запуска (все — вместе с --config):

Таблица 9

Флаг	Назначение
--setup-keys	Инициализировать каталог ключей и создать начальные ключи (однократно). Повторный запуск без --force завершается ошибкой, чтобы не уничтожить существующие ключи
--setup-keys --force	Пересоздать ключи в уже инициализированном каталоге (уничтожает старые ключи — данные, зашифрованные ими, станут невозстановимы!)

<code>--rotate-keys</code>	Ротация мастер-ключа с перешифрованием данных: новый ключ генерируется, данные перешифровываются, старые ключи удаляются только после успешного перешифрования (безопасно, при ошибке повторный запуск безопасен)
<code>--encrypt-legacy</code>	Однократная миграция: зашифровать open-text поля legacy-конфигурации доверенных сертификатов (protected_certs_config)
<code>--encrypt-legacy --force</code>	То же, но принудительно обрабатывает и значения, формально похожие на шифротекст, но не расшифровываемые активными ключами (осторожно: возможно безвозвратное перешифрование)

Все операции с ключами выполняются под блокировкой каталога ключей — параллельный запуск двух таких команд невозможен. Операции `--setup-keys/--rotate-keys/--encrypt-legacy` **не запускают HTTP-сервер**: выполнили команду — затем обычный запуск.

Важно: перед ротацией и любым обновлением версии делайте резервную копию каталога `protected.key_dir`. Без файла `0` расшифровать сохранённые зашифрованные данные невозможно.

4.4. Запуск сервера через бинарный файл

```
chmod +x ./taiga_porcini
./taiga_porcini --config=config.json
```

Путь к файлу конфигурации задаётся параметром `--config` обязательно. При отсутствии файла или ошибке в конфигурации сервис завершится с соответствующим сообщением об ошибке.

Примечание:

Бинарник поставки — `taiga_porcini`. Все операции по миграции данных встроены в сервис; отдельная утилита миграции не поставляется и не требуется.

- **Автоматический запуск** при загрузке системы
- **Автоматическое восстановление** при сбоях (перезапуск сервиса)
- **Централизованное управление** (старт, стоп, перезапуск)
- **Сбор логов** через journalctl
- **Мониторинг состояния** службы

4.5.1. Для Ubuntu / Debian / Astra Linux

1. Подготовка

Убедитесь, что сервис запускается вручную и работает корректно:

```
chmod +x ./taiga_porcini
./taiga_porcini --config=config.json
```

Создайте пользователя под сервис и установите ему пароль:

```
sudo adduser taiga
sudo passwd taiga
```

Создайте папку /opt/taiga и перенесите туда файл сервера и конфигурационный файл.

Примечание:

Проверьте, что у файла есть права на исполнение `chmod +x /opt/taiga/taiga_porcini`

Важно:

Перед первым запуском службы один раз инициализируйте хранилище мастер-ключей (от пользователя, под которым будет работать служба, чтобы каталог ключей ему принадлежал):

```
sudo -u taiga /opt/taiga/taiga_porcini --config /opt/taiga/config.json --setup-keys
```

Без инициализированного каталога `protected.key_dir` служба не запустится. Не забудьте включить этот каталог в резервное копирование.

Добавьте приложению доступ к портам ниже 1024

```
sudo setcap 'cap_net_bind_service=+ep' /opt/taiga/taiga_porcini
```

2. Создание файла службы

Создайте файл службы /etc/systemd/system/taiga.service:

```
sudo nano /etc/systemd/system/taiga.service
```

Вставьте следующую конфигурацию (замените пути на актуальные для вашей системы):

```
[Unit]
Description=Taiga Service
After=network.target
Wants=postgresql.service

[Service]
Type=simple
User=taiga
Group=taiga
WorkingDirectory=/opt/taiga
ExecStart=/opt/taiga/taiga_porcini --config /opt/taiga/config.json
Restart=always
RestartSec=10
StandardOutput=journal
StandardError=journal

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

Примечание:

Замените /opt/taiga на актуальный путь к директории с сервисом.

3. Активация и запуск службы

Перезагрузите менеджер служб:

```
sudo systemctl daemon-reload
```

Включите автоматический запуск при загрузке:

```
sudo systemctl enable taiga.service
```

Запустите службу:

```
sudo systemctl start taiga.service
```

4. Проверка состояния

Проверьте статус службы:

```
sudo systemctl status taiga.service
```

Ожидаемый вывод:

- taiga.service - Taiga Service
Loaded: loaded (/etc/systemd/system/taiga.service; enabled; vendor preset: enabled)
Active: active (running) since ...

5. Управление службой

Основные команды:

```
# Перезапуск службы
sudo systemctl restart taiga.service

# Остановка службы
sudo systemctl stop taiga.service

# Запуск службы
sudo systemctl start taiga.service

# Отключение автоматического запуска
sudo systemctl disable taiga.service

# Просмотр логов в реальном времени
sudo journalctl -u taiga.service -f

# Просмотр логов за сегодня
sudo journalctl -u taiga.service --since today

# Просмотр последних 100 строк логов
sudo journalctl -u taiga.service -n 100
```

Примечание про остановку:

systemctl stop отправляет процессу SIGTERM — сервис обрабатывает его штатно: корректно закрывает подключения к БД и кэшу (на корректное завершение отводится до 10 секунд). Принудительное завершение по таймауту systemd произойдёт только если процесс не завершился сам. То же относится и к Ctrl+C (SIGINT).

5. ЗАПУСК И ИСПОЛЬЗОВАНИЕ

5.1. Первоначальная настройка

Первоначальная настройка системы — это **однократный процесс инициализации**, который должен быть выполнен сразу после запуска сервиса. Пока система не инициализирована, любой запрос к веб-интерфейсу автоматически перенаправляется на мастер /setup. Этот этап включает:

1. Создание первой учетной записи администратора

2. Настройку глобальных параметров системы
3. Активацию лицензии (загрузка файла лицензии; подробнее — раздел «Лицензирование»)

При переходе на веб-сервер вы будете перенаправлены на `/setup`, откроется экран приветствия:

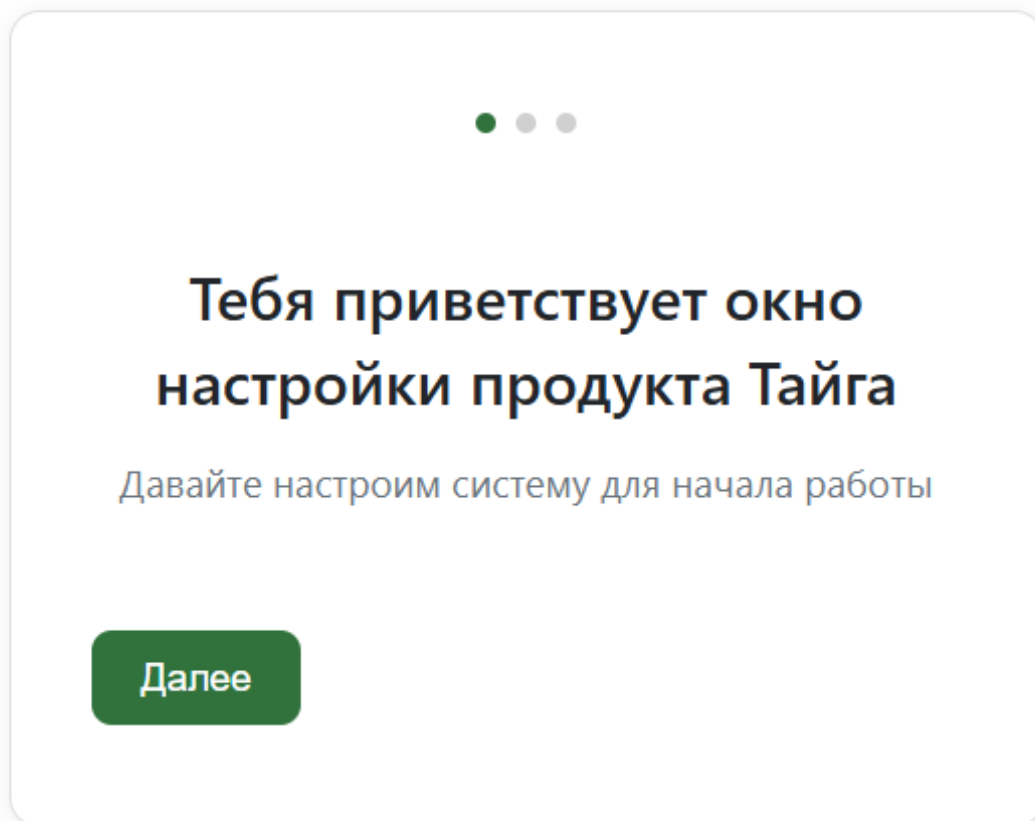
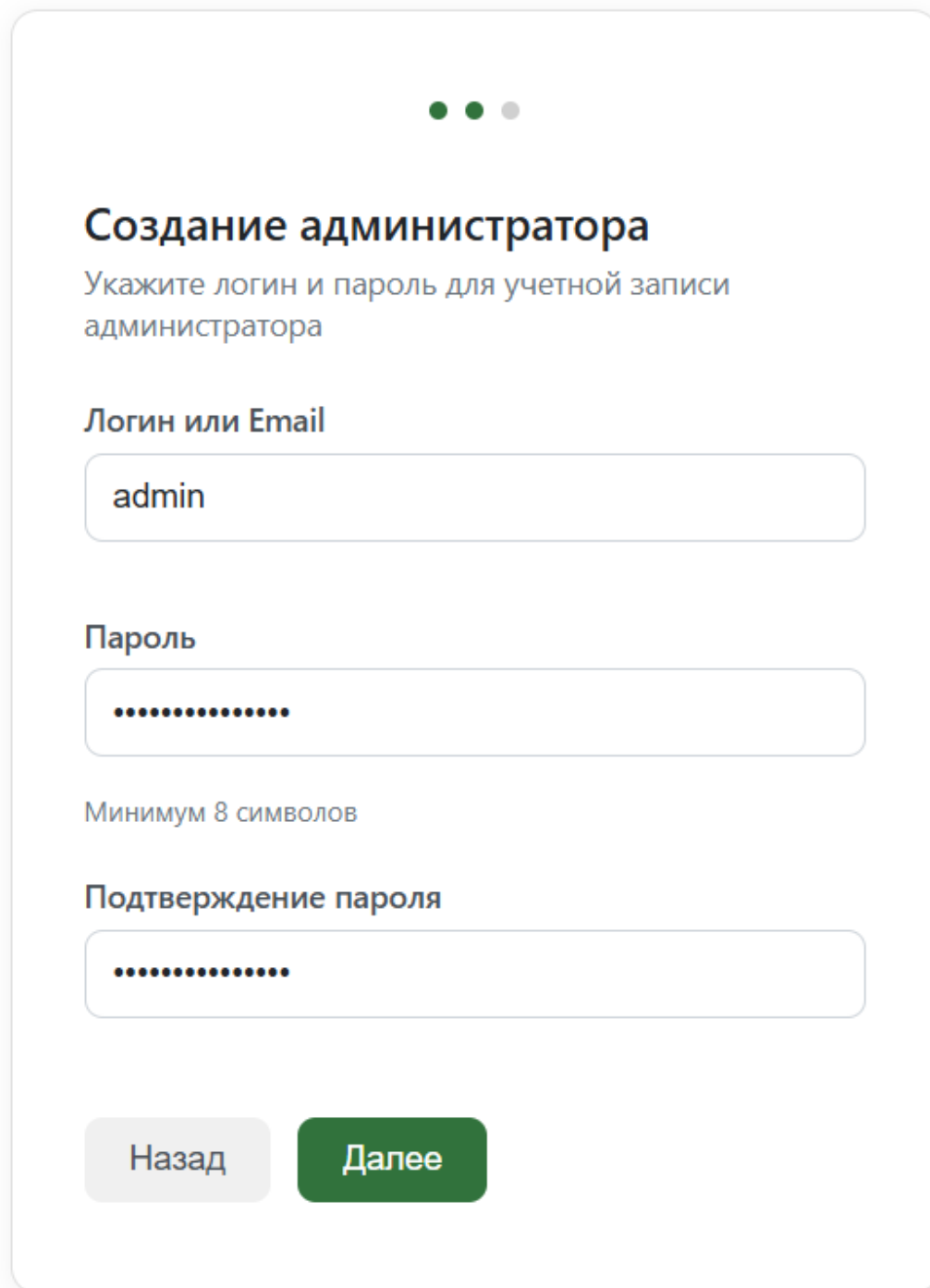


Рисунок 2 — Приветственное окно в браузере

Нажмите кнопку «Далее», чтобы перейти к заполнению данных администратора.



Создание администратора

Укажите логин и пароль для учетной записи администратора

Логин или Email

admin

Пароль

.....

Минимум 8 символов

Подтверждение пароля

.....

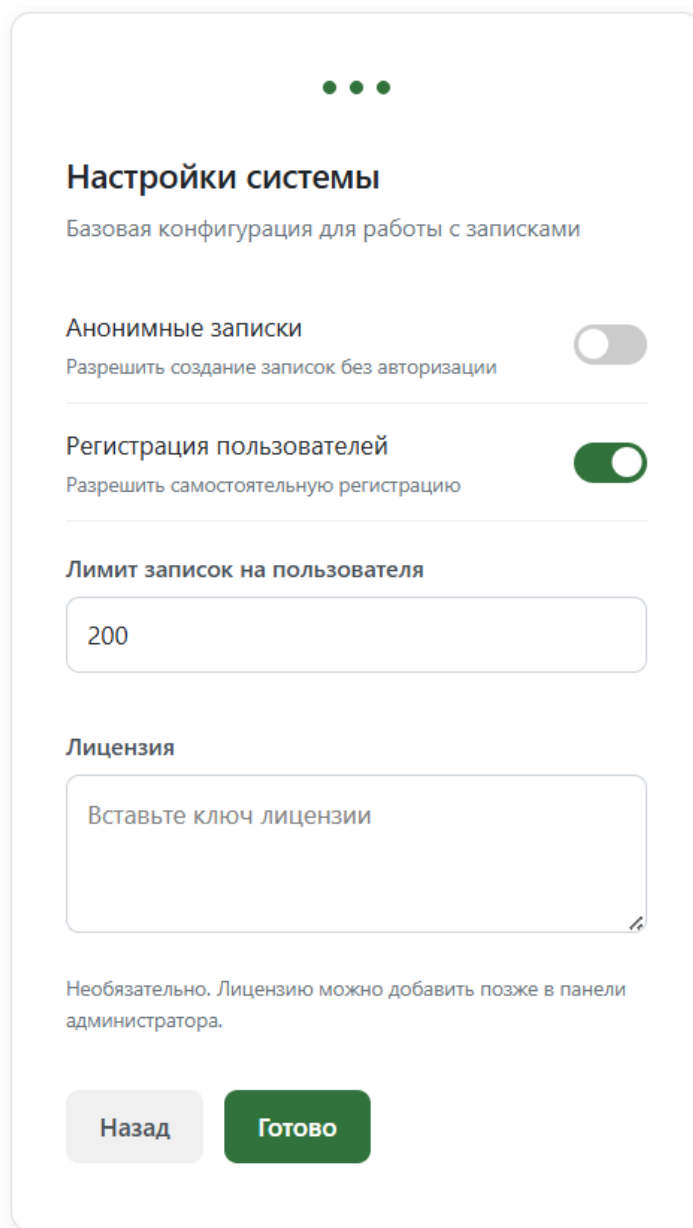
Назад Далее

Рисунок 3 — Создание администратора

Поля обязательны:

- **Email** — email администратора (используется как логин при входе в панель)
- **Пароль** — минимум 8 символов
- **Подтверждение пароля** — дублирование для исключения опечаток

После создания администратора вы задаёте общую политику системы.



Настройки системы

Базовая конфигурация для работы с записками

Анонимные записки

Разрешить создание записок без авторизации

Регистрация пользователей

Разрешить самостоятельную регистрацию

Лимит записок на пользователя

200

Лицензия

Вставьте ключ лицензии

Необязательно. Лицензию можно добавить позже в панели администратора.

Назад Готово

Рисунок 4 — Глобальные настройки системы

Таблица 10

Поле	Описание	Значение по умолчанию	Примечание
Разрешить анонимные записки	Могут ли незарегистрированные	Да	Влияет на безопасность: если

	пользователи создавать записи		отключено — только пользователи с аккаунтом могут инициировать записки
Разрешить публичную регистрацию	Разрешена ли самостоятельная регистрация новых пользователей	Да	Полезно для корпоративных решений (можно отключить и использовать только приглашения или синхронизацию LDAP)
Лимит записок на пользователя	Максимальное количество одновременно живых записок, которое может создать один пользователь	200	Учитывает как многоразовые, так и одноразовые записки. Сбрасывается при удалении записок

Совет:

- Для *частного использования* — оставьте все по умолчанию.
- Для *корпоративного сценария* — отключите публичную регистрацию и анонимные записки.

5.2. Демонстрационные сценарии

5.2.1. Тестовый сценарий: Создание и чтение записки

После инициализации сервиса откроется главная страница приложения **Тайга**. Здесь вы можете создать свою первую записку, воспользовавшись удобным интерфейсом.

Откройте главную страницу

После авторизации вы попадёте на экран создания новой записки. Этот экран содержит все необходимые инструменты для шифрования, добавления файлов и выбора метода защиты данных.

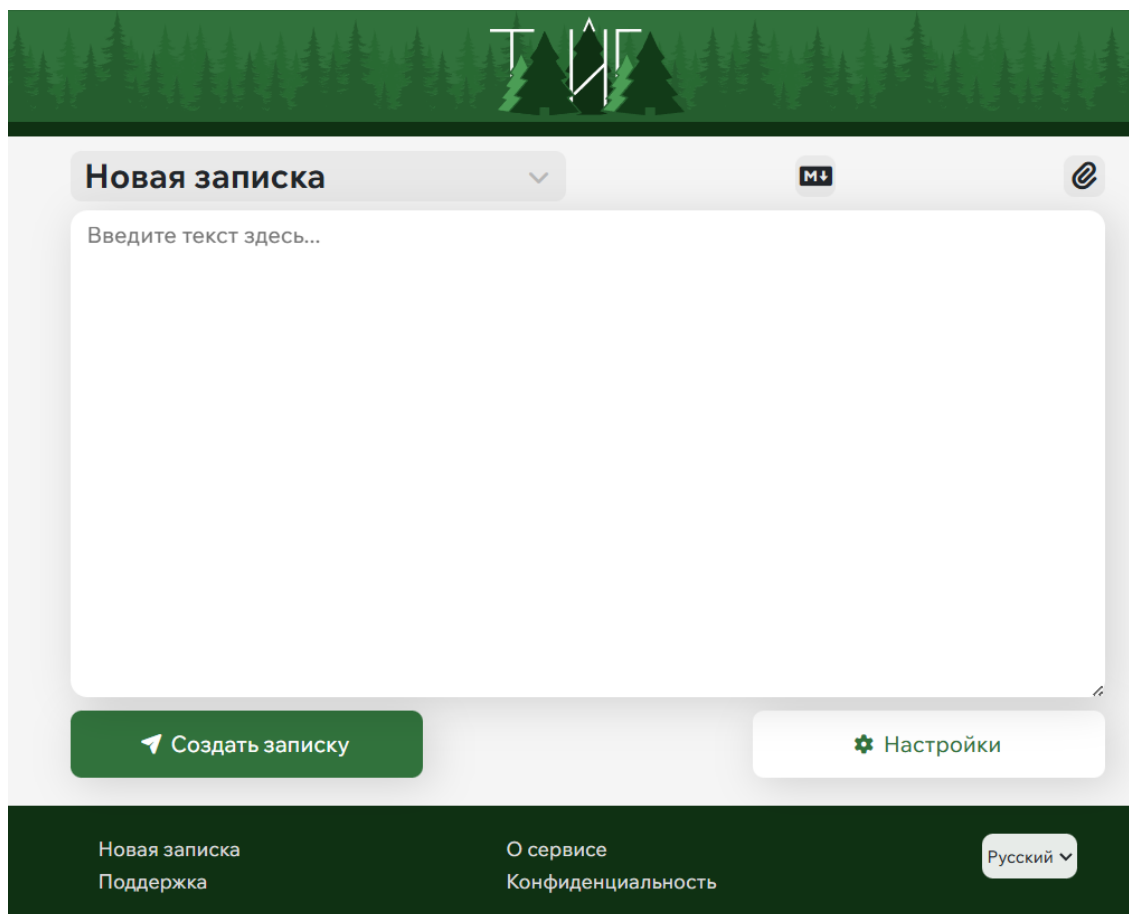


Рисунок 5 — Главная страница

Введите содержимое записки

В центральной части экрана расположен блок **ввода текста**. Здесь можно написать обычный текст или использовать Markdown для форматирования.

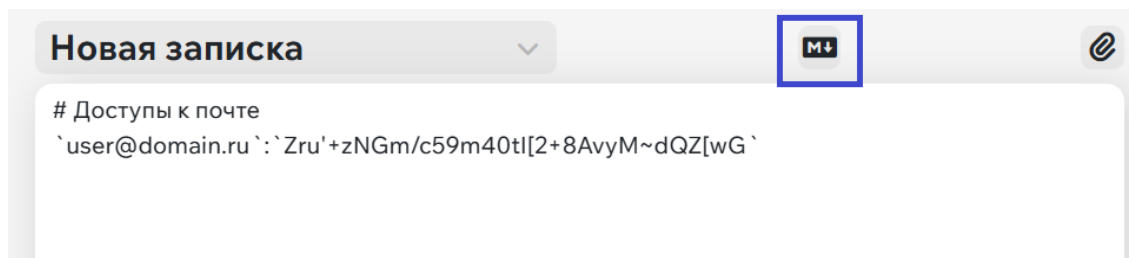


Рисунок 6 — Кнопка для включения Markdown разметки

При включенном Markdown можно включить предпросмотр нажав на иконку.

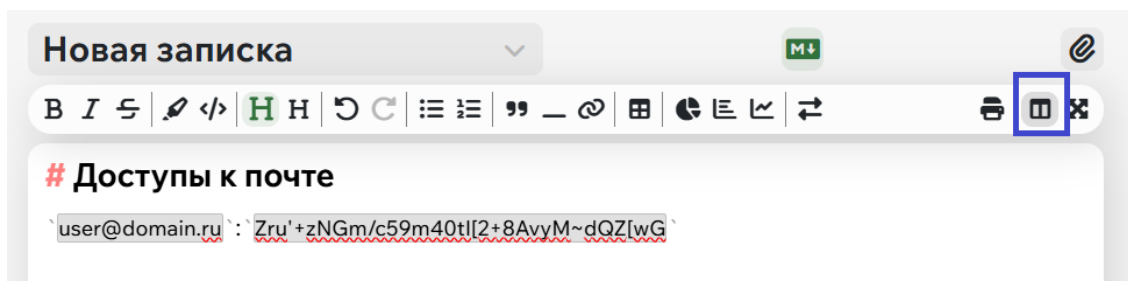


Рисунок 7 — Кнопка для включения предпросмотра Markdown разметки

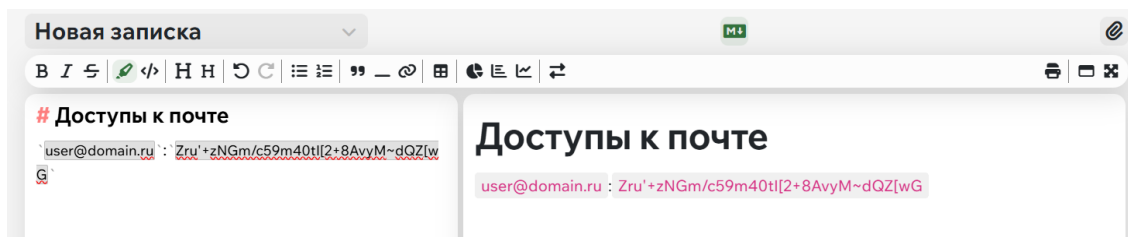


Рисунок 8 — Включенный предпросмотр

Настройте записку

Для большинства сценариев достаточно стандартных настроек, но вы можете настроить записку под себя, для этого откройте блок «Настройки»



Рисунок 9 — Кнопка для открытия настроек

В открывшемся поле:

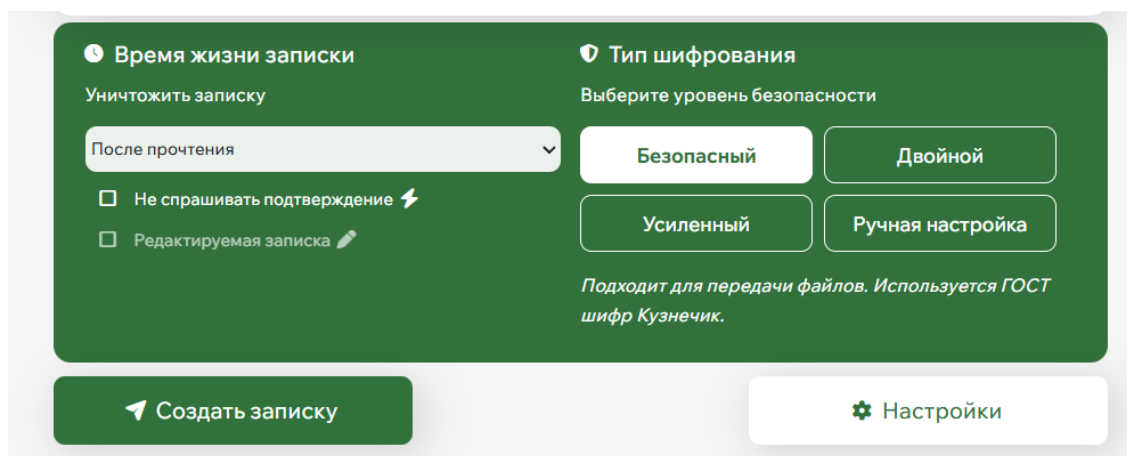


Рисунок 10 — Меню настроек записки

Время жизни записки. Выберите, как долго ваша записка будет доступна:

- **После прочтения** (по умолчанию). Удалится сразу после прочтения, одноразовая.
- **15 минут, 1 час, 7 дней** и т.д.

Метод шифрования. Здесь можно выбрать уровень защиты:

- **Безопасный** (рекомендуется по умолчанию): используется шифр Кузнечик (ГОСТ). Подходит для передачи файлов.
- **Двойной**: более высокий уровень защиты. Сначала сообщение шифруется шифром Кузнечик, затем AES.
- **Усиленный**: записка шифруется с помощью XOR, ключ больше или равен длине записки. Режим подходит только для небольших текстов.
- **Ручная настройка**: позволяет выбрать конкретные параметры шифрования.

Создайте записку

Нажмите кнопку «Создать записку».



Рисунок 11 — Кнопка для создания записки

После успешного создания появится ссылка на вашу записку.

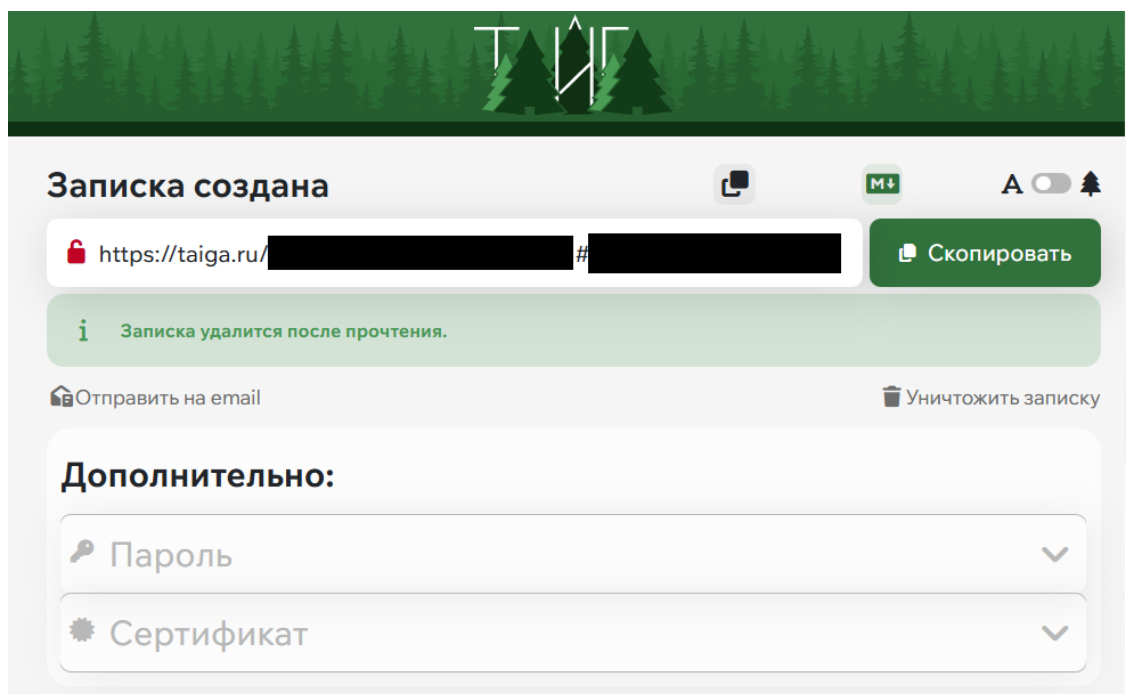


Рисунок 12 — Страница со ссылкой на записку

Примечание:

Отправьте эту ссылку получателю, она не будет отображаться снова.

Дополнительные действия

После создания записки доступны дополнительные действия:

- **Дублировать записку** – возвращает на странице редактирования, с ранее заполненной запиской. Позволяет поправить текст и создать ещё одну записку.
- **Копировать ссылку** - чтобы передать её другому человеку.
- **Уничтожить записку** - немедленно удалить запись (необратимо).
- **Дополнительно:**. Если вы хотите дополнительно зашифровать ключ в ссылке – используйте вкладки:
 - Пароль
 - Сертификат (импортированные или доверенные сертификаты)

6. СПЕЦИФИЧЕСКИЕ ОСОБЕННОСТИ

6.1. Миграции базы данных

Миграции встроены в бинарник сервиса и автоматически применяются к базе данных при каждом запуске — отдельная схема не требуется, все объекты создаются в схеме public. Отдельного запуска утилиты миграции не нужно. Убедитесь, что БД готова к подключению, а у пользователя БД есть права на создание таблиц и последовательностей в схеме public.

6.2. Шифрование и ключи

Тайга использует библиотеку WellspringJS, которая в свою очередь реализует следующие стандарты:

- Кузнечик (ГОСТ 34.12-2018) - основное шифрование
- Стрибог (ГОСТ 34.11-2018) - для генерации ключей
- ГОСТ 34.10-2018 - для цифровых подписей

Примечание:

Шифрование в браузере реализовано через WellspringJS, который интегрируется и работает только во фронтенде. Сервер не хранит и не обрабатывает открытый текст, а также не хранит ключей.

6.3. Разграничение доступа

Система поддерживает несколько режимов дополнительной защиты ссылок:

Таблица 11

Режим	Описание
password	Ключ в ссылке дополнительно шифруется с помощью исключающего или ключа и PBKDF2 с помощью хэш-функции Стрибог (ГОСТ 34.11-2018)
certificate	Ключ в ссылке дополнительно шифруется с помощью эфемерного закрытого ключа и открытого ключа из сертификата

	пользователя, вырабатываемого с помощью функции КЕГ (Р 1323565.1.020-2018) общего ключа для шифрования ссылки.
--	--

Примечание:

Режимы могут быть включены через UI при создании записки.

6.4. Лицензирование

Сервис «Тайга» работает только при наличии **активной лицензии**. Пока лицензия отсутствует, не активирована или истекла, блокирующий механизм ограничивает доступ приложения:

- доступны: страница входа `/admin/login`, страница управления лицензией `/admin/license`, мастер первоначальной настройки `/setup`, CAPTCHA (`/api/v1/captcha`), эндпоинты аутентификации (`/api/v1/auth/*`), проверка лицензии (`/api/v1/license/validate`), `healthcheck` (`/api/v1/healthcheck`) и статика интерфейса;
- всё остальное (включая страницы записок и основную часть API) отдаёт страницу-заглушку «требуется лицензия» / «лицензия истекла» (HTTP 403) или JSON-ошибку `license_required` / `license_expired`;
- криптографические WASM-модули без активной лицензии также заблокированы — без них полноценная работа с записками невозможна.

При запуске сервиса лицензия, сохранённая в БД, **проверяется криптографически** (подпись Ed25519 формата JWS). Лицензия с невалидной подписью трактуется как отсутствующая — сервис переходит в режим «требуется лицензия».

Активация лицензии выполняется администратором через веб-страницу `/admin/license` (загрузка файла лицензии `.jws`) или в мастере первоначальной настройки; программно — через API `PUT /api/v1/admin/license` (требуется JWT и роль `admin`).

Формат лицензии, её выпуск и проверка описаны в отдельном документе «Описание технических средств активации, выпуска, распространения и управления лицензионными ключами СОЗ „Тайга“» (код НМА-001-LICENSE). Лицензионный файл поставляется правообладателем; для активации достаточно загрузить файл лицензии `.jws` через раздел «Лицензия» административной панели.

7. СИСТЕМНОЕ ЛОГИРОВАНИЕ

Подробная информация о настройке логирования представлена в разделе Конфигурационные файлы (секция logging).

По умолчанию логи выводятся в консоль (stdout/stderr). Для сохранения в файл используйте перенаправление:

```
./taiga_porcini --config=config.json > /var/log/taiga.log 2>&1
```

7.1. Уровни логирования

Таблица 12

Уровень	Описание
trace	Трассировочные сообщения для глубокой отладки
debug	Для разработки и диагностики внутренних операций
info	Информационные сообщения о нормальной работе системы
warn	Предупреждения, не требующие немедленного вмешательства
error	Ошибки обработки запросов или задач
fatal	Критические ошибки, приводящие к завершению приложения

Совет. Для продуктовой среды рекомендуется использовать `info` или `warn` с JSON-форматом (`encoding: json`) для удобного парсинга в системах мониторинга.

8. УСТРАНЕНИЕ НЕПОЛАДОК: ТИПИЧНЫЕ ОШИБКИ И ИХ РЕШЕНИЕ

8.1. Ошибки при запуске сервера

8.1.1. Ошибка: «Port is already in use» (порт занят)

Причина:

Порт, указанный в конфигурационном файле (config.json), уже используется другим процессом.

Решение:

1. Проверьте использование порта:

```
lsof -i :8080
```

2. Остановите конфликтующий процесс или измените порт в config.json (значение — строка с двоеточием, не число):

```
{  
  "http": {  
    "port": ":8081"  
  }  
}
```

Примечание:

Сервер выдает ошибку *listen tcp :8080: bind: address already in use*. Порт может быть изменён в конфиге или через переменную окружения с именем `http.port` (имя переменной = путь ключа в конфиге, см. раздел «Переменные окружения»). Переменной `TAIGA_HTTP_PORT` не существует. Указание числа вместо строки (`"port": 8081`) — невалидная конфигурация, сервис её не примет.

8.2. Ошибки при подключении к PostgreSQL

8.2.1. Ошибка: «could not connect to server» (не подключается к БД)

Причина:

Неверные параметры соединения с БД, либо сервер PostgreSQL не запущен.

Решение:

1. Проверьте конфигурационный файл:

```
{
  "db": {
    "host": "localhost",
    "port": "5432",
    "user": "taiga_user",
    "pwd": "secure_password",
    "database": "taiga_db",
    "sslmode": "disable"
  }
}
```

2. Убедитесь, что PostgreSQL работает:

```
systemctl status postgresql
```

3. Проверьте доступность БД:

```
psql -U taiga_user -d taiga_db
```

Примечание:

Ошибки подключения к PostgreSQL выводятся в журнал сервиса. Убедитесь, что пользователь БД имеет права на базу данных и параметры секции `db` корректны.

8.3. Ошибки инициализации хранилища ключей и конфигурации

8.3.1. Ошибка: «protected.key_dir is required» (не задан каталог ключей)

Причина: в `config.json` отсутствует секция `protected` или пуст `key_dir`.

Решение: добавьте обязательную секцию `protected` с путём `key_dir` (см. описание конфигурации).

8.3.2. Ошибка: «key repository directory does not exist: ... — run --setup-keys first» (хранилище ключей не инициализировано)

Причина: каталог `protected.key_dir` не существует — перед первым запуском не была выполнена инициализация ключей.

Решение:

```
./taiga_porcini --config=config.json --setup-keys
./taiga_porcini --config=config.json
```

8.3.3. Ошибка: «jwt secret is required» / «jwt secret must be at least 32 characters»

Причина: в `jwt.secret` задан слишком короткий секрет (менее 32 символов).

Решение: сгенерируйте криптостойкий секрет (`openssl rand -base64 32`) и запишите в `jwt.secret`. Если поле не задавать вовсе, сервис сгенерирует случайный секрет автоматически — но тогда токены будут сброшены после перезапуска, для продуктовой среды секрет задавайте явно.

8.4. Ошибки в браузере

8.4.1. Ошибка: "WASM modules not loaded" (не загружены WASM модули)

Причина:

WebAssembly модули для шифрования (`cryptoTaiga_wasm_128.wasm`, `cryptoTaiga_wasm_64.wasm`) не доступны в браузере.

Решение:

1. Обновите браузер до современного
2. Убедитесь, что в браузере включена поддержка WebAssembly
3. Проверьте, что активирована лицензия: без активной лицензии криптографические WASM-модули блокируются (HTTP 403) и шифрование в браузере работать не будет

Примечание:

Ускоренное оконечное шифрование и Markdown-разметка зависят от WASM. Если модули не загружены, пользователь не сможет использовать полный функционал сервиса.

8.5. Ошибки при работе с CAPTCHA

8.5.1. Ошибка: "captcha validation failed" (CAPTCHA не проходит)

Причина:

Неверная конфигурация CAPTCHA; либо сложность выросла из-за частых запросов с вашего IP (при значении по умолчанию сложность равна $\text{start_complexity} + N / \text{step_complexity}$, где N — число не протухших CAPTCHA-запросов с этого IP за последний час).

Решение:

1. Проверьте параметры CAPTCHA в config.json:

```
{
  "captcha": {
    "start_complexity": 6,
    "step_complexity": 3
  }
}
```

2. Убедитесь, что время не менялось. Обновите страницу и повторите попытку отправки записи.
3. Если запросов с одного IP было много — подождите истечения часа и повторите попытку (записи CAPTCHA хранятся в кэше 1 час).

8.6. Ошибки при работе с LDAP

8.6.1. Ошибка: «LDAP connection failed» (не подключается к LDAP)

Причина:

LDAP-сервер недоступен или конфигурация неверна.

***Важно!** LDAP-настройки не задаются в config.json. Они настраиваются через административный интерфейс после запуска сервиса.*

Решение:

1. Зайдите в админ-панель сервиса
2. Перейдите в раздел «Провайдеры идентификации»
3. Проверьте настройки LDAP-подключения:
 - URL LDAP-сервера (ldaps:// или ldap://)
 - DN администратора и пароль
 - Базовый DN для поиска пользователей
4. Убедитесь, что сервер доступен:

```
ldapsearch -H ldaps://example.com -b "" -s base "(objectclass=*)"
```

8.7. Ошибки при работе с кэшем (Redis)

8.7.1. Ошибка: "cache initialization failed" (кэш не инициализирован)

Причина:

Конфигурационные параметры кэша некорректны.

Решение:

1. Проверьте config.json (обратите внимание на типы: port — строка, database — число):

```
{
  "cache": {
    "host": "localhost",
    "port": "6379",
    "user": "",
    "pwd": "",
    "database": 0,
    "sslmode": "disable",
    "ca_file": ""
  }
}
```

1. Для TLS-подключения используйте sslmode: require (без проверки сертификата) либо verify-ca/verify-full с указанием ca_file (PEM-сертификат CA).
2. Убедитесь, что Redis доступен:

```
redis-cli ping
```

Примечание:

Встроенная in-memory кэш-реализация (silverbox) используется **только когда секция cache в config.json полностью отсутствует**. Это не является ошибкой, но может снизить производительность при высокой нагрузке.

Если же секция cache задана, Redis обязателен: при его недоступности сервис **не запустится** с ошибкой init redis cache (вложенная connect to redis). Чтобы вернуть использование встроенного in-memory кэша, удалите секцию cache из конфигурации.

8.8. Ошибки в логировании

8.8.1. Ошибка: «logs not output» (логи не выводятся)

Причина:

Неверный уровень логирования или отсутствие файла для сохранения.

Решение:

1. Запустите с детальным логированием:

```
./taiga_porcini --config=config.json
```

2. Проверьте вывод в консоль или указанные в конфиге файлы логов.

Совет. Используйте уровень *debug* для диагностики проблем:

```
{  
  "logging": {  
    "level": "debug",  
    "encoding": "console",  
    "output_paths": ["stdout"]  
  }  
}
```

9. ПОДДЕРЖКА

9.1. Контакты

E-mail: support@taiga.ru

Телеграм: https://t.me/taiga_help_chat

По ним можно:

- Получить техническую помощь
- Отправить баг-репорты
- Связаться с командой разработчиков