Общество с ограниченной ответственностью «Инновационный центр безопасности»
ИНН: 7724395003/ КПП: 773101001

ИНСТРУКЦИЯ ПО УСТАНОВКЕ И ЭКСПЛУАТАЦИИ

СОЗ «ТАЙГА»

Версия документа: 1.01 от 22.03.2026

Код: НМА-001-INSTRUCT

г. Москва, 2026 г.

121596, г. Москва, ул. Толбухина, д. 9, к. 1, пом.1/П support@taiga.ru

СОДЕРЖАНИЕ

1. ТЕРМИНЫ И СОКРАЩЕНИЯ

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

Оконечное (сквозное) шифрование (end-to-end encryption, E2EE) — оконечное 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 расшифрования 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 обмена записками «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.

3.2. Зависимости

3.2.1. База данных

- **PostgreSQL** ≥ 13:
  - Основное хранилище данных (записки, пользователи, сертификаты).
  - Схема taiga обязательна для создания.
  - Требует прав на создание таблиц и триггеров.

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 CAPTCHA.
  - Без Redis приложение работает, но с сниженной производительностью.

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 и другим критически важным настройкам.

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"
  },
  "http": {
    "port": ":80",
    "tls": {
      "cert": "cert.pem",
      "key": "key.pem"
    }
  },
  "options": {
    "with_admin": true
  },
  "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
  }
}
```

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 (обязательно создайте схему `taiga` перед запуском)

http — настройки HTTP-сервера

Таблица 2

Параметр	Тип	Обязательный	Описание
`port`	`string`	да	Порт HTTP-сервера (например, `:80`, `:8080`, `:443`)
`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). При каждой неудачной попытке решения сложность растёт: start_complexity + N × step_complexity, где N — номер попытки. Это предотвращает 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 `12`

> ***Совет.*** *Чем выше bcrypt_cost, тем медленнее проверка паролей, но тем устойчивее система к брутфорсу. Для продуктовой среды рекомендуется 12–14.*

options — опциональные параметры

Таблица 6

Параметр	**Тип**	**Обязательный**	**Описание**
`with_admin`	`bool`	нет	Включить начальный экран настройки администратора. По умолчанию `true`

4.2.4. Проверка конфигурации

Перед запуском убедитесь, что:

1. База данных `taiga_db` создана, пользователь имеет права на создание таблиц и триггеров.
2. В `jwt.secret` не меньше 32 символов.

3. Для HTTPS-режима (порт `:443`) указаны `tls.cert` и `tls.key`, и файлы существуют.
4. Порт HTTP-сервера не занят другими приложениями.

> ***Примечание:***
> *Ошибка в конфигурации приведёт к ошибке could not connect to server, invalid JWT secret, или listen tcp: address already in use 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. Запуск сервера через бинарный файл

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

Если файл конфигурации не указан, сервис попытается загрузить его по умолчанию из `./config.json`. При отсутствии файла или ошибке в конфигурации сервис завершится с соответствующим сообщением об ошибке.

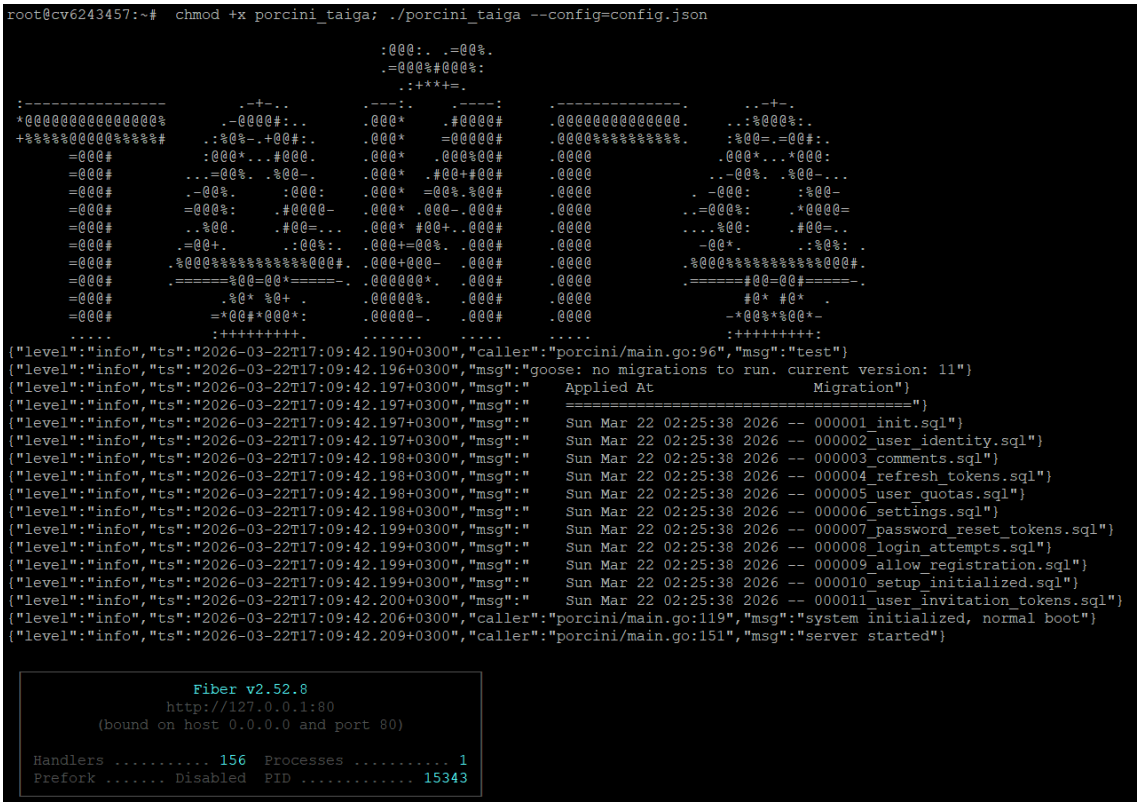

Рисунок 1 — Пример успешного запуска сервера

После запуска сервиса проверьте доступность HTTP-интерфейса:

```
curl http://localhost:8080/ # должен вернуть 200 OK
```

Готово! Теперь вы можете перейти к первоначальной настройке (созданию администратора и системных параметров) или сразу создать первую записку.

4.4. Запуск через системные службы (systemd)

Для продуктовой среды рекомендуется запускать сервис через системную службу **systemd**. Это обеспечивает:

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

4.4.1. Для Ubuntu / Debian / Astra Linux

1. Подготовка

Убедитесь, что сервис запускается вручную и работает корректно:

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

Создайте пользователя под сервис и установите ему пароль:

```
sudo adduser porcini
sudo passwd porcini
```

Создайте папку /opt/taiga в домашней директории пользователя и перенесите туда файл сервера и конфиогурационный файл.

> ***Примечание:***
>
> *Проверьте, что у файла есть права на исполнение chmod +x /opt/taiga/porcini*

Добавьте приложению доступ к портам ниже 1024

```
sudo setcap 'cap_net_bind_service=+ep' /opt/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=porcini
Group=porcini
WorkingDirectory=/opt/porcini/taiga
ExecStart=/opt/porcini/taiga/porcini --config /opt/porcini/taiga/config.json
Restart=always
RestartSec=10
StandardOutput=journal
StandardError=journal
```

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

> ***Примечание:***
>
> *Замените /opt/porcini/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
```

5. ЗАПУСК И ИСПОЛЬЗОВАНИЕ

5.1. Первоначальная настройка

Первоначальная настройка системы — это **однократный процесс инициализации**, который должен быть выполнен сразу после запуска сервиса. Этот этап включает:

1. Создание первой учетной записи администратора
2. Настройку глобальных параметров системы

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

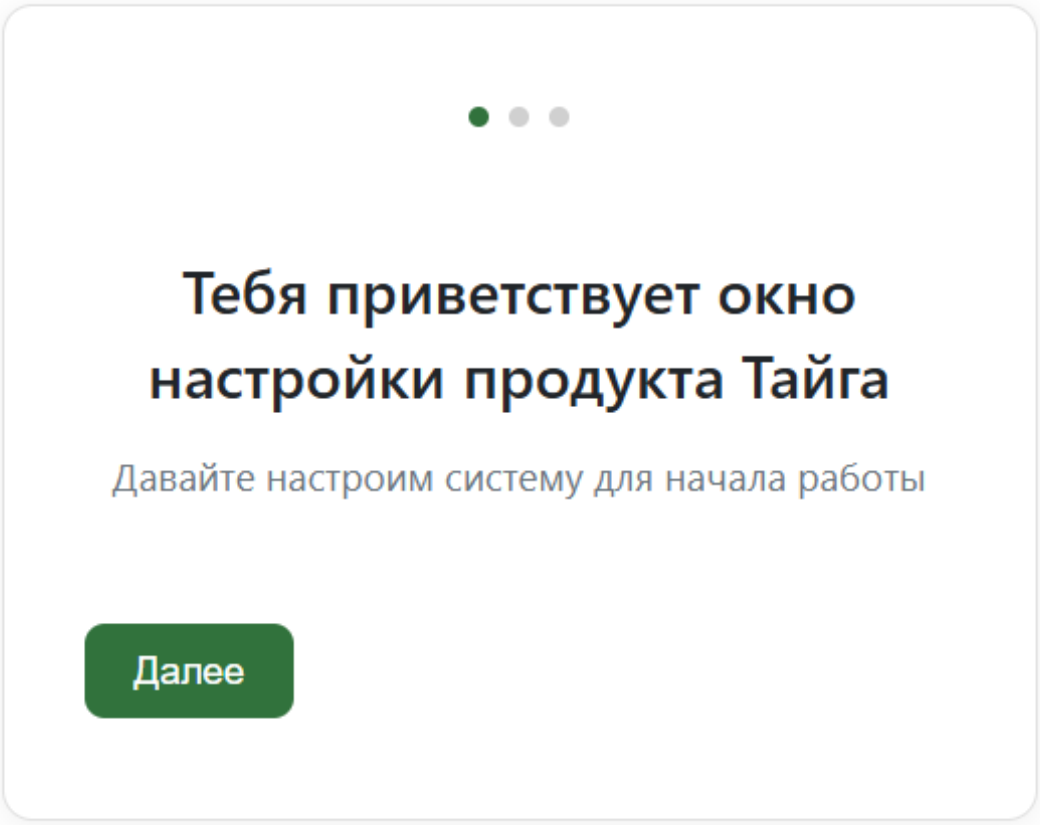

Рисунок 2 — Приветственное окно в браузере

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

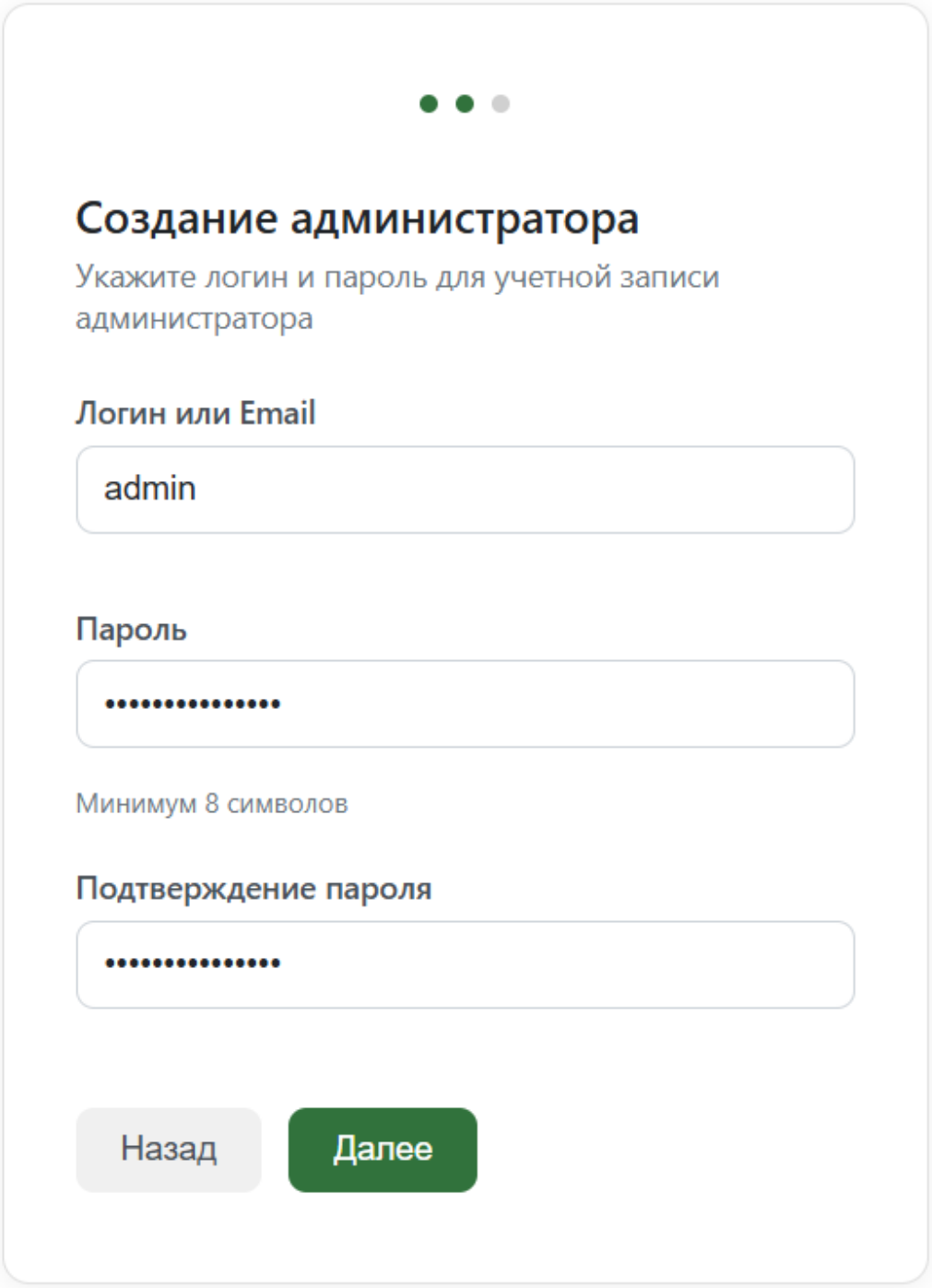

Рисунок 3 — Создание администратора

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.

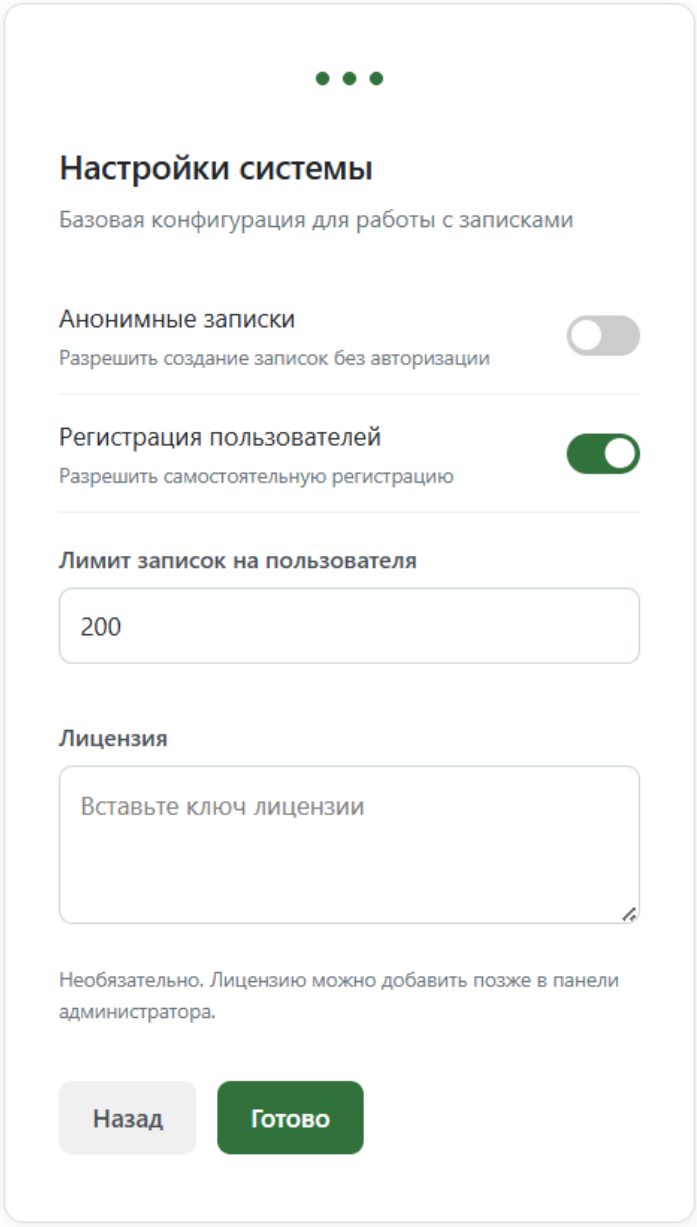

Рисунок 4 — Глобальные настройки системы

Таблица 7

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.

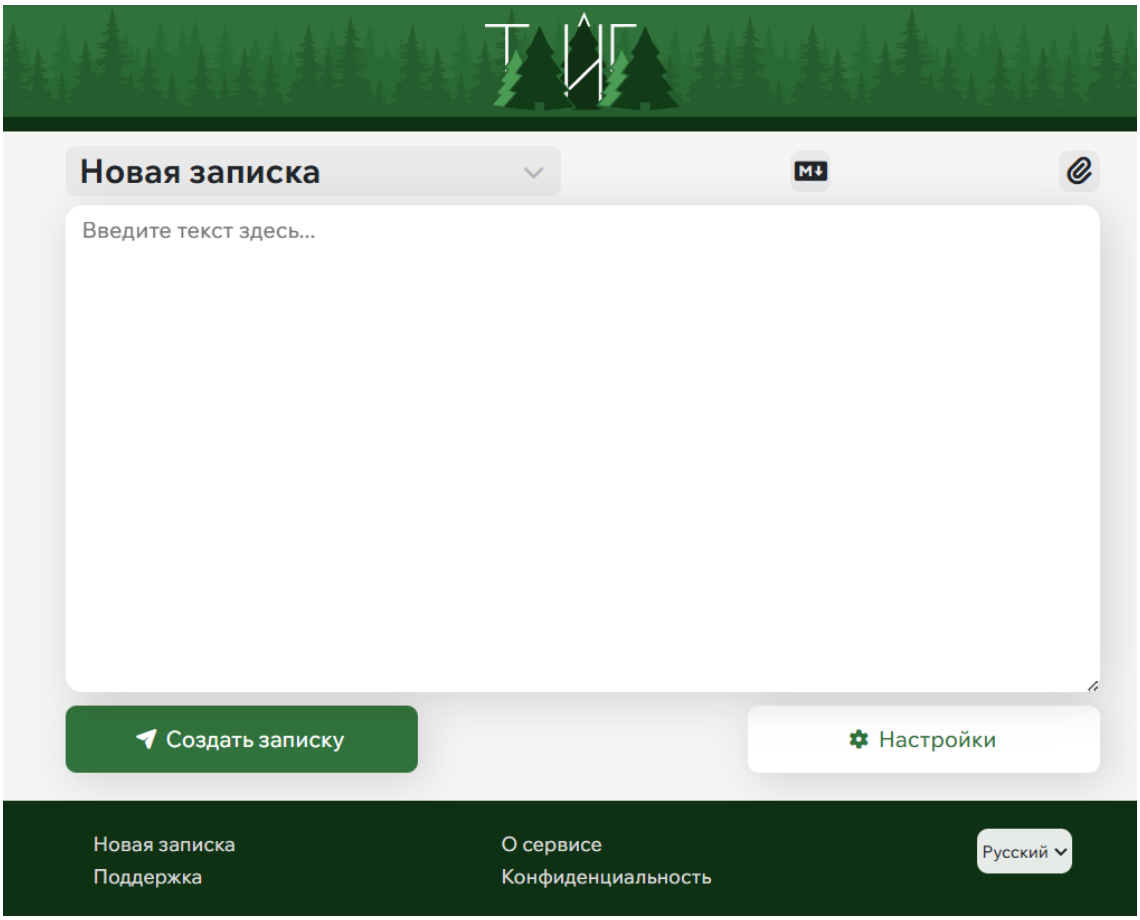

Рисунок 5 — Главная страница

Введите содержимое записки

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

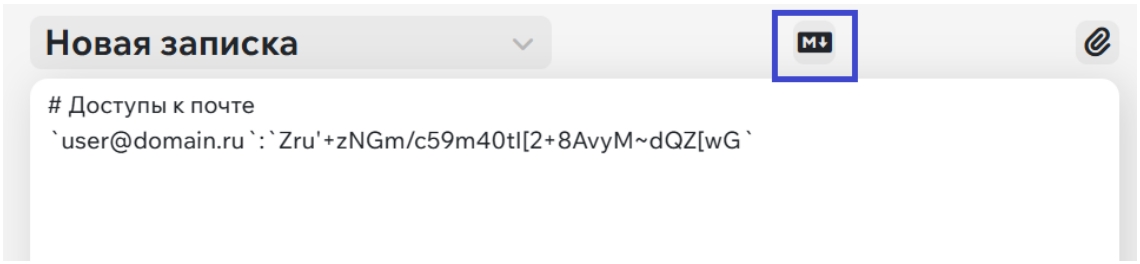

Рисунок 6 — Кнопка для включения Markdown разметки

При включенном Markdown можно включить предпросмотр нажав на иконку.

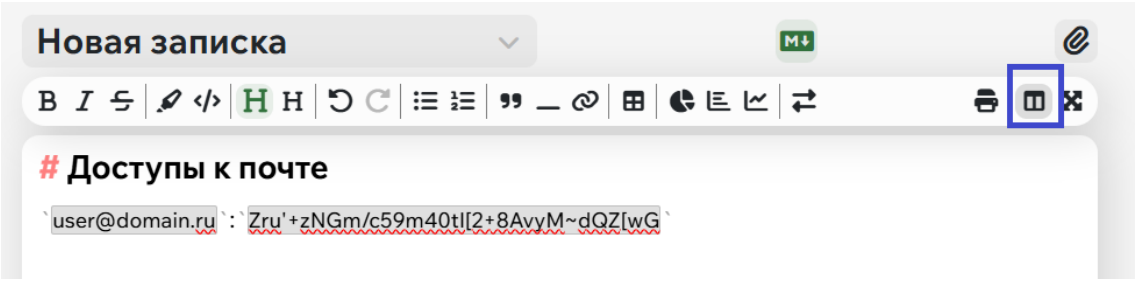

Рисунок 7 — Кнопка для включения предпросмотра Markdown разметки

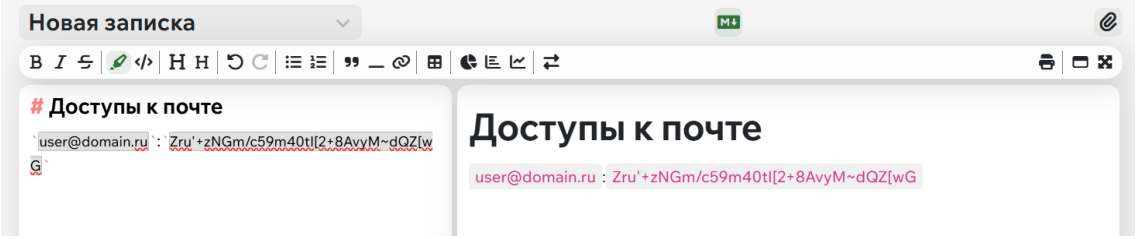

Рисунок 8 — Включенный предпросмотр

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:

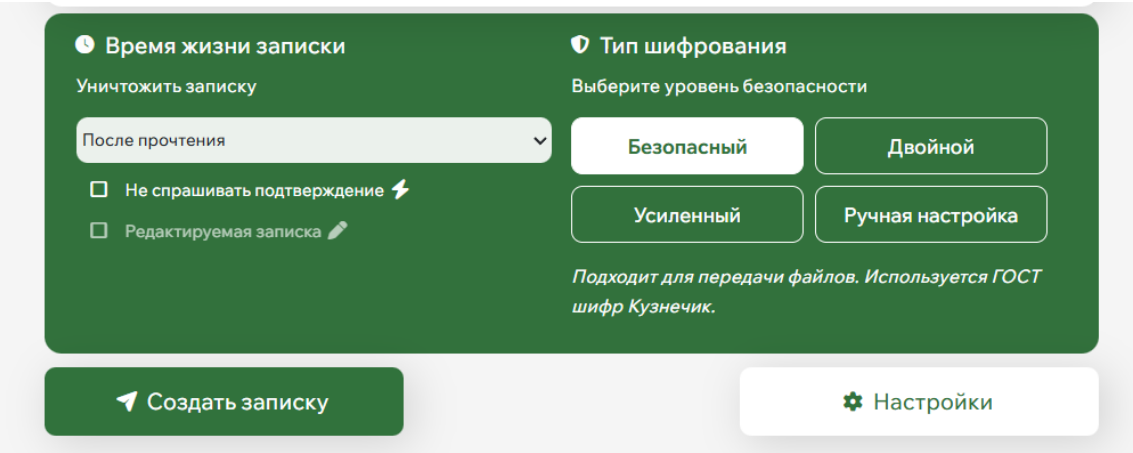

Рисунок 10 — Меню настроек записки

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:

- **Простое**: автоматический выбор режима шифра Кузнечик. (рекомендуется по умолчанию).
- **Двойное / Максимальное**: 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».**

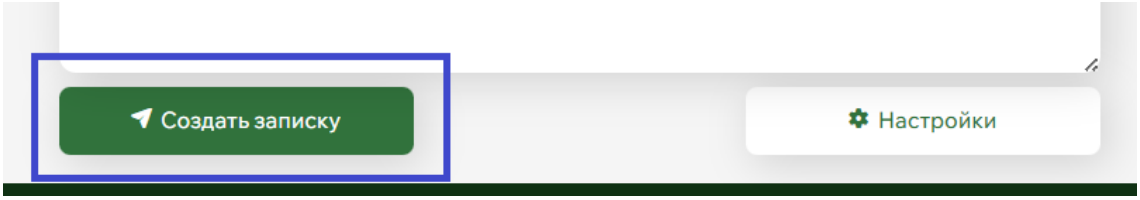

Рисунок 11 — Кнопка для создания записки

После успешного создания появится ссылка на вашу записку.

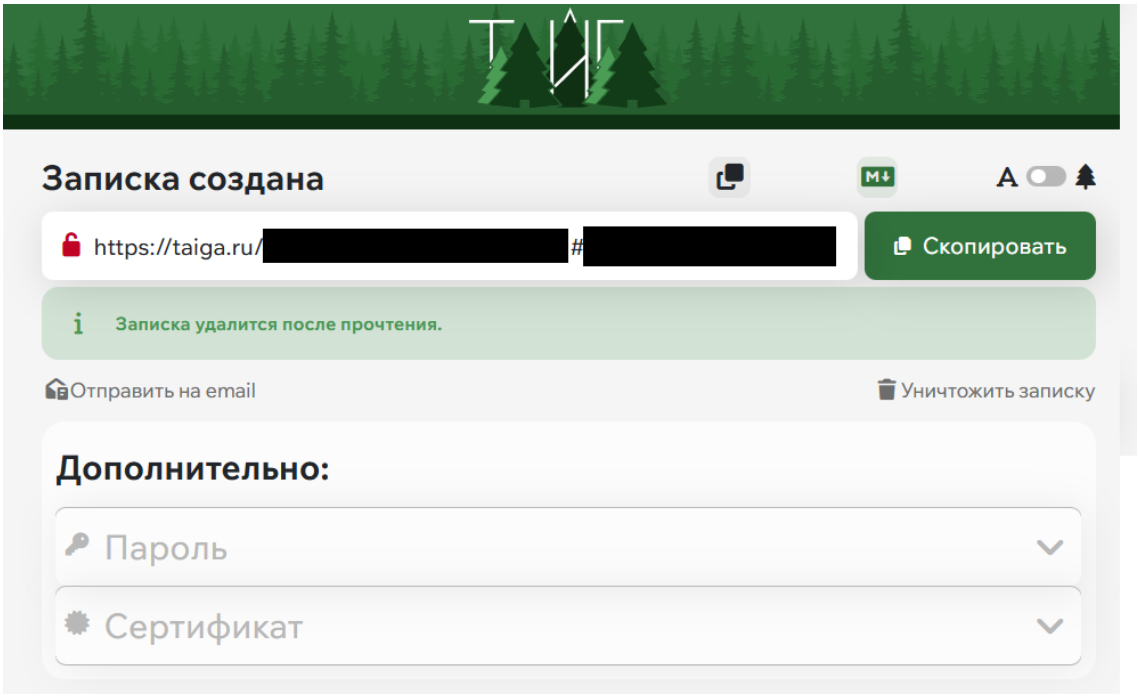

Рисунок 12 — Страница со ссылкой на записку

> ***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.
- **Удалить сейчас** - немедленно удалить запись (необратимо).
- **Зашифровать дополнительно**. Если вы хотите дополнительно зашифровать ключ в ссылке – используйте:
  - Пароль
  - Импортированные или доверенные сертификаты

6. СПЕЦИФИЧЕСКИЕ ОСОБЕННОСТИ

6.1. Миграции базы данных

Миграции автоматически запускаются при запуске новой версии программы и применяются к PostgreSQL схеме taiga. Убедитесь, что БД готова к подключению.

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 8

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, вырабатыва с помощью функции KEG (Р 1323565.1.020-2018) общий

	ключ для шифрования ссылки.

> ***Примечание:***
>
> *Режимы могут быть включены через UI при создании записки.*

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:

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

7.1. Уровни логирования

Таблица 9

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`:

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

> ***Примечание:***
>
> *Сервер выдает ошибку listen tcp :8080: bind: address already in use. Порт может быть изменён через переменную окружения TAIGA_HTTP_PORT или в конфиге.*

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:***
>
> *Ошибка может быть обернута через errors.Wrap() в storage/pg/connection.go.*
>
> *Убедитесь, что пользователь имеет права на базу данных.*

8.3. Ошибки в браузере

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

Причина:

WebAssembly модули для шифрования (`128.wasm`, `64.wasm`) не доступны в браузере.

Решение:

1. Обновите браузер до современного

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

8.4. Ошибки при работе с CAPTCHA

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

Причина:

Неверная конфигурация CAPTCHA или отсутствие библиотеки для вычисления хешей.

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 записки.

8.5. Ошибки при работе с LDAP

8.5.1. 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=*)"
```

8.6. Ошибки при работе с кэшем (Redis)

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

Причина:

Конфигурационные параметры кэша некорректны.

Решение:

1. Проверьте `config.json`:

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

2. Убедитесь, что Redis доступен:

```
redis-cli ping
```

> ***Примечание:***
>
> *Если секция cache отсутствует или Redis недоступен, сервис автоматически использует встроенную in-memory кэш-реализацию (silverbox). Это не является ошибкой, но может снизить производительность при высокой нагрузке.*

8.7. Ошибки в логировании

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

Причина:

Неверный уровень логирования или отсутствие файла для сохранения.

Решение:

1. Запустите с детальным логированием:

```
./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