Обновление и обслуживание модулей личного кабинета

Данная инструкция описывает процедуру обновления установленных модулей личного кабинета. Включает в себя как мониторинг текущего состояния стенда, так и пошаговое руководство по выполнению обновлений. Актуальная информация о списке установленных модулей и их версиях доступна на странице «Администрирование | Модули».

Содержание

⚠ Важно: Общие рекомендации и принципы

Для обеспечения стабильной и предсказуемой эксплуатации системы необходимо соблюдать следующие принципы:

- **Регулярное обновление** — выполняйте обновления периодически, не допуская большого отставания в версиях. Это снижает риски и делает переход на новые версии безболезненным.
- **Резервное копирование** — своевременно создавайте резервные копии базы данных и файлов приложения перед любыми изменениями.
- **Тестирование** — всегда тестируйте обновления на выделенном тестовом окружении перед применением в production.
- **Проверка совместимости** — убедитесь в совместимости версий PHP, ядра TANDEM LK и сторонних зависимостей.
- **Постоянный контроль** — информационная система требует регулярного технического обслуживания и мониторинга для предотвращения сбоев и несанкционированных ситуаций.

Примечание от команды Личных коммуникационных сервисов ТАНДЕМ: Мы тщательно тестируем каждый релиз, стабилизируем сборки и стараемся предусмотреть возможные проблемные сценарии. Однако в силу сложности распределённых систем и разнообразия клиентских окружений **гарантия 100 % безошибочной работы невозможна**. Мы — команда профессионалов, но мы тоже не застрахованы от ошибок. Ваша бдительность, соблюдение рекомендаций и своевременная обратная связь — ключ к надёжной эксплуатации системы.

⛔ Запрещённые действия

Нарушение данных правил влечёт высокий риск потери данных, нарушения безопасности или отказа системы.

- НИКОГДА не устанавливайте APP_DEBUG=true в production-окружении.
- НИКОГДА не удаляйте миграции из базы данных вручную.
- НИКОГДА не останавливайте worker-процессы очередей без крайней необходимости и согласования.
- НИКОГДА не обновляйте production без предварительного резервного копирования.
- НИКОГДА не используйте драйвер sync для очередей в production.
- НИКОГДА не вносите изменения в код напрямую на production-сервере.

Если у вас есть вопросы, замечания или пожелания — обращайтесь в нашу команду через систему Service Desk (SD). Мы оперативно реагируем на все обращения, особенно — на инциденты критического уровня.

🔧 Требования к эксплуатации

Общие обязанности Заказчика

Заказчик (клиент) несёт полную ответственность за корректную эксплуатацию системы в соответствии с техническими требованиями и рекомендациями, изложенными в настоящей документации.

Инфраструктура и оборудование

Заказчик обязан обеспечить:

- **Серверное оборудование**:

    - наличие сервера с достаточными ресурсами (CPU, RAM, дисковое пространство);
    - соответствие минимальным системным требованиям;
    - резервирование и отказоустойчивость (при необходимости);
    - регулярное техническое обслуживание.
- **Сетевая инфраструктура**:
    - стабильное интернет-соединение с достаточной пропускной способностью;
    - настройку DNS для доменного имени;
    - своевременное обновление SSL-сертификатов;
    - настройку и поддержку правил файрвола;
    - доступ к внешним сервисам: github.com, packagist.org и др.
- **Системное ПО**:
    - ОС: Ubuntu LTS / CentOS / AltLinux / AstraLinux (актуальная LTS-версия);
    - веб-сервер: Nginx (рекомендуется) или Apache;
    - PHP = 7.4;
    - СУБД: MySQL или MariaDB;
    - Redis;
    - Composer и другие необходимые инструменты.

Администрирование

Заказчик обязан:

- Назначить квалифицированного администратора, ознакомленного с документацией.
- Обеспечить администратору и разработчику (при необходимости) полный доступ к системе.
- Выполнять мониторинг:
    - ежедневно — доступность и базовые метрики;
    - еженедельно — анализ логов и производительности;
    - ежемесячно — аудит конфигурации и безопасности.
- Обеспечивать безопасность: регулярные обновления ОС, контроль доступа, мониторинг аномалий, соблюдение

политики паролей и доступа

Резервное копирование

- **Обязательно**:
  - ежедневное резервное копирование БД;
  - резервная копия файлов приложения и .env перед каждым обновлением;
  - резервирование загруженных файлов и конфигурации сервера.
- Регулярно проверять возможность восстановления из резервных копий.

Обновления и обслуживание

Заказчик обязан:

- **Выполнять обновления в соответствии с процедурами:**
  - следовать инструкциям по обновлению из настоящей документации;
  - создавать резервные копии перед началом обновления;
  - тестировать обновления на выделенном тестовом окружении (при его наличии);
  - применять обновления только в режиме обслуживания (php artisan down);
  - проверять работоспособность всех ключевых сценариев после завершения обновления.
- **Поддерживать актуальность системы:**
  - регулярно (но с осторожностью) обновлять зависимости Личного кабинета — только после проверки совместимости;
  - оперативно применять обновления безопасности — в течение 72 часов с момента выпуска;
  - обновлять системные пакеты операционной системы в рамках утверждённого графика (рекомендуется — еженедельно).

Конфигурация и настройка

Заказчик обязан:

- **Правильно настроить окружение:**
  - корректная установка и настройка системных пакетов, требуемых для работы личного кабинета;
  - точная и безопасная настройка файла .env;
  - установка параметров APP_ENV=production и APP_DEBUG=false в production-окружении;
  - настройка подключений к базе данных и Redis (валидные хост, порт, учётные данные);
  - настройка параметров очередей (QUEUE_CONNECTION) и кэширования (CACHE_DRIVER, REDIS_CACHE_DB и др.).
- **Обеспечить правильные права доступа:**
  - настройка прав на файлы и директории в соответствии с требованиями фреймворка Laravel:
    - 755 — для директорий,
    - 644 — для файлов,
    - 775 — для storage/ и bootstrap/cache/ (владелец — пользователь веб-сервера);
  - контроль доступа к критическим файлам (.env, SSH-ключи, конфиги);
  - регулярная проверка и аудит прав доступа (рекомендуется — ежемесячно).

Мониторинг и отчётность

Заказчик обязан:

- **Вести мониторинг системы:**
  - регулярный мониторинг всех компонентов (серверы, БД, Redis, очереди, HTTP-статусы);
  - анализ результатов мониторинга (метрики нагрузки, ошибки, латентность);
  - оперативное реагирование на предупреждения и ошибки;

    - ведение журнала изменений и инцидентов с указанием даты, причины, действий и результата.
- **Предоставлять информацию при запросах:**
    - результаты мониторинга при возникновении проблем;
    - фрагменты логов системы (особенно storage/logs/laravel.log, nginx/error.log);
    - информацию о недавно выполненных обновлениях, изменениях конфигурации и развёртываниях.

Рекомендации для заказчика

Для обеспечения стабильной и предсказуемой работы системы рекомендуется:

- **Обучить администратора:**
    - ознакомить с настоящей документацией;
    - провести обучение по работе с системами мониторинга;
    - обучить процедурам обновления и резервного копирования.
- **Настроить автоматизацию:**
    - автоматическое резервное копирование БД и файлов;
    - автоматический запуск проверок мониторинга и отправка алертов;
    - автоматическая ротация логов (например, через logrotate).
- **Вести внутреннюю документацию:**
    - журнал изменений и обновлений;
    - журнал инцидентов и способов их устранения;
    - документацию по специфическим настройкам и кастомизациям.
- **Организовать тестовое окружение:**
    - для проверки обновлений перед развёртыванием в production;
    - для тестирования изменений конфигурации;
    - для обучения и аттестации администраторов.
- **Планировать техническое обслуживание:**

- выделять регулярные окна обслуживания для обновлений (например, еженедельно в нерабочее время);
- проводить плановые проверки системы (безопасность, резервное копирование, мониторинг);
- выполнять профилактические работы (очистка диска, актуализация ОС, аудит логов).

Взаимодействие с разработчиком

Заказчик обязан:

- **Информировать о проблемах:**
  - предоставлять детальное и структурированное описание инцидента;
  - прикладывать результаты мониторинга и фрагменты логов (с указанием временных меток);
  - описывать точные шаги для воспроизведения проблемы (включая входные данные и ожидаемый/фактический результат).
- **Соблюдать установленные процедуры:**
  - следовать инструкциям разработчика при устранении инцидентов;
  - не вносить изменения в исходный код без предварительного письменного согласования;
  - согласовывать плановые обновления и крупные изменения в инфраструктуре (если это предусмотрено условиями сопровождения).

Соблюдение обязанностей, изложенных в настоящем разделе, является критически важным для обеспечения стабильной, безопасной и корректной работы системы личного кабинета.

Мониторинг стенда

Цель мониторинга — подтверждение работоспособности всех компонентов системы в едином окружении.

Основные проверки

1. Валидация конфигурации

```
php composer.phar validate --no-check-publish
```

2. Проверка .env

- APP_KEY — задан.
- APP_ENV и APP_DEBUG — соответствуют окружению:
  - **Production**: APP_ENV=production, APP_DEBUG=false
  - **Development / Testing**: APP_ENV=develop, APP_DEBUG=true
- QUEUE_CONNECTION = database — задан для работы через БД.
- **Redis** — подключение настроено:
  - REDIS_CLIENT=predis (или phpredis)
  - REDIS_HOST=127.0.0.1 (или IP/хост Redis-сервера)
  - REDIS_PORT=637
  - REDIS_PASSWORD=null (если пароль не требуется) или актуальный пароль
- **База данных** — корректные параметры подключения:
  - DB_CONNECTION=mysql
  - DB_HOST=127.0.0.1
  - DB_PORT=3306
  - DB_DATABASE=tandem_lk
  - DB_USERNAME=root
  - DB_PASSWORD=root
  - *Рекомендуется использовать отдельного пользователя БД с ограниченными привилегиями, а не root.*
- **Почта** — настроены параметры отправки:
  - MAIL_MAILER=smtp
  - Заданы: MAIL_HOST, MAIL_PORT, MAIL_USERNAME,

MAIL_PASSWORD, MAIL_ENCRYPTION, MAIL_FROM_ADDRESS.

- **Интеграция с НСИ** (если установлен модуль nsi-client):
  - NSI_CLIENT_SRC_SUBSYSTEM_CODE="tandem"
  - NSI_CLIENT_DST_SUBSYSTEM_CODE="nsi"
  - NSI_CLIENT_LOGIN="tandem"
  - NSI_CLIENT_PWD="password" (замените на актуальный пароль)
  - NSI_CLIENT_ADAPTER_URL=http://localhost:8180 (актуальный URL адаптера НСИ)
  - NSI_CLIENT_FORCE_QUEUE=true (рекомендуется для production)
- **Прочие параметры** — проверьте наличие и корректность всех переменных, требуемых используемыми модулями.

Важно:

- После изменения .env выполните php artisan config:clear, чтобы сбросить кэш конфигурации.

3. Права доступа

Убедитесь, что веб-сервер имеет права на запись в:

- storage/
- bootstrap/cache/
- (при необходимости) vendor/

4. Ресурсы сервера

```
# Uptime (время работы системы)

cat /proc/uptime | awk '{print int($1/86400) " дней"}'

# Количество ядер

nproc

# Linux
```

```
free -h

free -m | grep Mem | awk '{print "Использовано: " $3 "MB / " $2 "MB (" int($3/$2*100) "%)"}'

# Диск

df -h

df -h | grep -E '^/dev/|^tmpfs' | awk '{print $1 ": " $3 "/" $2 " (" $5 ")"}'

# Статистика сети

ip -s link show

ip addr show
```

5. Доступ к внешним ресурсам

```
# Через nc (netcat)

nc -z -v -w5 registry.npmjs.org 80 443

nc -z -v -w5 github.com 80 443

nc -z -v -w5 packagist.org 80 443

nc -z -v -w5 tandem.gitlab.yandexcloud.net 22 80 443

# Через telnet

telnet registry.npmjs.org 80

telnet registry.npmjs.org 443

telnet github.com 80

telnet github.com 443
```

```
telnet api.github.com 80

telnet api.github.com 443

telnet packagist.org 80

telnet packagist.org 443

telnet repo.packagist.org 80

telnet repo.packagist.org 443

telnet tandem.gitlab.yandexcloud.net 80

telnet tandem.gitlab.yandexcloud.net 443

telnet tandem.gitlab.yandexcloud.net 22
```

6. Состояние служб

Убедитесь, что запущены и работают:

- Nginx/Apache
- PHP-FPM
- MySQL/MariaDB
- Redis
- Supervisor (для воркеров)

7. Миграции и БД

```
# Статус миграций
php artisan migrate:status

# Поиск неприменённых миграций
php artisan migrate:status | grep "Pending"

# Размер БД
php artisan tinker --execute="
\$result = DB::select('SELECT ROUND(SUM(data_length + index_length) / 1024 / 1024, 2) AS size_mb FROM information_schema.tables WHERE table_schema = DATABASE()');
echo \$result[0]->size_mb . ' MB';"
```

8. Подключение к Redis

```
php artisan tinker --execute="
use Illuminate\Support\Facades\Redis;
try {
    Redis::ping();
    echo 'Подключение успешно';
} catch (Exception \$e) {
    echo 'Ошибка: ' . \$e->getMessage();
}"
```

9. Логи

```
# Размер логов
du -sh storage/logs
find storage/logs -type f -size +100M

# Количество файлов
ls -1 storage/logs | wc -l
```

10. CRON и очереди

```
# CRON
crontab -l | grep "schedule:run"

# Супервизор
sudo supervisorctl status

# Проверка через процессы

# Поиск процессов очередей

sudo pgrep -f "queue:work"

sudo ps aux | grep "queue:work"

# Кол-во задач в очереди

php artisan tinker --execute="

\$queues = ['default', 'base', 'system', 'event_notification',
'delivery_notification', 'bus_message_delivery'];
```

```
foreach (\$queues as \$queue) {

    \$count = Queue::size(\$queue);

    echo \$queue . ': ' . \$count . PHP_EOL;

}"
```

Интерпретация результатов

Уровень	Критерии
Критический	• Диск > 90% • Службы не запущены (MySQL, Redis, Nginx) • Нет подключения к БД • Критические ошибки в storage/logs/
Предупреждение	• Диск 80–90% • Load Average > кол-ва CPU • RAM > 90% • Неприменённые миграции • APP_DEBUG=true в production • Устаревшие зависимости
Норма	• Диск < 80% • Load Average < CPU • RAM < 80% • Все службы работают

Обновление стенда

Обновление приложения может выполняться как средствами личного кабинета так и вручную.

Процесс обновления через интерфейс (автоматический)

Для выполнения обновления средствами личного кабинета требуется перейти в раздел «Администрирование» — «Модули» и нажать кнопку «Обновить модули».

Порядок обновления средствами личного кабинета (выполняется автоматически, действий администратора не требует):

1. Система переходит в режим обслуживания (php artisan down).
2. Выполняется composer update tandem/* --no-dev.
3. Очищаются скомпилированные файлы.
4. Обновляется автозагрузчик.
5. Применяются миграции: php artisan migrate --force.
6. Публикуются ресурсы: php artisan tandem:publish.
7. Назначаются права: php artisan tandem:permissions.
8. Пересобираются фронтенд-ресурсы: npm install && npm update && npm rebuild && npm run prod.
9. Очищается кеш: php artisan route:clear && php artisan cache:clear && php artisan view:clear && php artisan config:clear
10. Запускаются команды php artisan tandem:update:{module} для модулей с кастомной логикой обновления.
11. Обновляется информация о доступных версиях.
12. Устанавливаются обязательные модули (если есть).
13. Система возвращается в рабочий режим (php artisan up).

Процесс выполняется асинхронно в очереди system. При большом отставании в кодовой базе может потребоваться перезапуск Supervisor перед повторным запуском обновления.

Если система ведет себя нестандартно информация фиксируется в разделе «Администрирование» — «Модули» — «Журнал».

В зависимости от возникшей ошибки требуется устранить системные ошибки, которые возникли в результате обновления (например: недоступность репозиториев, проблема с правами) и выполнить повторно обновление или выполнить ручное обновление.

Ручное обновление (рекомендуется при сбое автоматического)

1. Режим обслуживания и остановка воркеров

```
php artisan down
sudo service supervisor stop или sudo service supervisord stop

# 2. Обновление зависимостей
composer update --no-dev

# 3. Очистка скомпилированных файлов
php artisan clear-compiled

# 3. Обновление автозагрузчика Composer
composer dump-autoload

# 4. Миграции
php artisan migrate --force

# 5. Очистка кэшей
php artisan route:clear
php artisan cache:clear
php artisan view:clear
php artisan config:clear
php artisan optimize:clear

# 6. Публикация ресурсов модулей
php artisan tandem:publish

# 7. Обновление прав доступа
php artisan tandem:permissions

# 8. Перезапуск Супервизора
sudo service supervisor restart или sudo service supervisord
restart

# 9. Обновление npm-зависимостей и сборка
npm install
npm update
npm rebuild
npm run prod

# 10. Обновление отдельных модулей (опционально)

#  Для  каждого  модуля,  у  которого  есть  команда
```

```
tandem:update:{module_name}
# Примеры:

# php artisan tandem:update:base
# php artisan tandem:update:journal
# php artisan tandem:update:schedule
# и т.д.
# Проверить доступные команды обновления:
# php artisan list | grep "tandem:update:"

# 11. Обновление информации о доступных версиях (опционально)
php artisan tandem:update:available-version

# 12. Поднятие приложения
php artisan up
```

Важные замечания

- **Резервная копия** — обязательна перед началом.
- **Пользователь** — команды выполняйте от владельца приложения.
- **Git-доступ** — требуется SSH-ключ для git@tandem.gitlab.yandexcloud.net.
- **Время** — 5–15 минут в зависимости от объёма изменений.

Управление передачей пакетов из ЛК в НСИ

Для реализации двустороннего обмена между ЛК и УНИ реализован модуль nsi-client, который позволяет передавать в УНИ информацию об изменении данных в ЛК. Процесс отправки данных происходит в фоновом режиме на сервере ЛК. За отправку отвечает системное средство supervisor .

Принудительная остановка и запуск отправки пакетов

Для управления передачей необходимо зайти в терминал сервера,

на котором установлен ЛК от имени администратора (root) и выполнить команду supervisorctl status. Данная команда отображает состояние всех фоновых процессов. В данном случае нас интересуют процессы bus-message-delivery-worker:<номер_процесса>.

```
root:~# supervisorctl status
adapter-send:adapter-send_00                                    RUNNING   pid 1307839, uptime 1 day, 16:20:09
bus-message-delivery-worker:bus-message-delivery-worker_00      RUNNING   pid 1451731, uptime 0:13:18
bus-message-delivery-worker:bus-message-delivery-worker_01      RUNNING   pid 1451732, uptime 0:13:18
bus-message-delivery-worker:bus-message-delivery-worker_02      RUNNING   pid 1451733, uptime 0:13:18
bus-message-delivery-worker:bus-message-delivery-worker_03      RUNNING   pid 1451734, uptime 0:13:18
bus-message-delivery-worker:bus-message-delivery-worker_04      RUNNING   pid 1451735, uptime 0:13:18
bus-message-delivery-worker:bus-message-delivery-worker_05      RUNNING   pid 1451736, uptime 0:13:18
bus-message-delivery-worker:bus-message-delivery-worker_06      RUNNING   pid 1451737, uptime 0:13:18
bus-message-delivery-worker:bus-message-delivery-worker_07      RUNNING   pid 1451738, uptime 0:13:18
delivery-notification-worker:delivery-notification-worker_00    RUNNING   pid 1307848, uptime 1 day, 16:20:09
event-notification-worker:event-notification-worker_00          RUNNING   pid 1307849, uptime 1 day, 16:20:09
io-chat-worker:io-chat-worker_00                                RUNNING   pid 1307850, uptime 1 day, 16:20:09
lk-base:lk-base_00                                              RUNNING   pid 1307851, uptime 1 day, 16:20:09
lk-base:lk-base_01                                              RUNNING   pid 1307852, uptime 1 day, 16:20:09
lk-base:lk-base_02                                              RUNNING   pid 1307853, uptime 1 day, 16:20:09
lk-base:lk-base_03                                              RUNNING   pid 1307854, uptime 1 day, 16:20:09
lk-base:lk-base_04                                              RUNNING   pid 1307855, uptime 1 day, 16:20:09
lk-base:lk-base_05                                              RUNNING   pid 1379038, uptime 19:26:23
lk-base:lk-base_06                                              RUNNING   pid 1307857, uptime 1 day, 16:20:09
lk-base:lk-base_07                                              RUNNING   pid 1307858, uptime 1 day, 16:20:09
lk-report:lk-report_00                                          RUNNING   pid 1307859, uptime 1 day, 16:20:09
lk-report:lk-report_01                                          RUNNING   pid 1307860, uptime 1 day, 16:20:09
lk-report:lk-report_02                                          RUNNING   pid 1307861, uptime 1 day, 16:20:09
lk-report:lk-report_03                                          RUNNING   pid 1307862, uptime 1 day, 16:20:09
lk-report:lk-report_04                                          RUNNING   pid 1307863, uptime 1 day, 16:20:09
lk-report:lk-report_05                                          RUNNING   pid 1307864, uptime 1 day, 16:20:09
lk-report:lk-report_06                                          RUNNING   pid 1307865, uptime 1 day, 16:20:09
lk-report:lk-report_07                                          RUNNING   pid 1307867, uptime 1 day, 16:20:09
lk-system-base:lk-system-base_00                                RUNNING   pid 1307868, uptime 1 day, 16:20:09
```

Для остановки отправки пакетов в НСИ необходимо выполнить команду supervisorctl stop bus-message-delivery-worker:* После этого отправка будет приостановлена, что можно увидеть на странице *Администрирование — Модули — Настройки главного модуля — Очередь.* После остановки в очереди будут копиться пакеты с очереди bus-message-delivery.

```
root:~# supervisorctl stop bus-message-delivery-worker:*
bus-message-delivery-worker:bus-message-delivery-worker_01: stopped
bus-message-delivery-worker:bus-message-delivery-worker_02: stopped
bus-message-delivery-worker:bus-message-delivery-worker_00: stopped
bus-message-delivery-worker:bus-message-delivery-worker_03: stopped
bus-message-delivery-worker:bus-message-delivery-worker_04: stopped
bus-message-delivery-worker:bus-message-delivery-worker_05: stopped
bus-message-delivery-worker:bus-message-delivery-worker_06: stopped
bus-message-delivery-worker:bus-message-delivery-worker_07: stopped
```

Для запуска отправки пакетов в НСИ необходимо выполнить команду supervisorctl start bus-message-delivery-worker:*

```
root:~# supervisorctl start bus-message-delivery-worker:*
bus-message-delivery-worker:bus-message-delivery-worker_00: started
bus-message-delivery-worker:bus-message-delivery-worker_01: started
bus-message-delivery-worker:bus-message-delivery-worker_02: started
bus-message-delivery-worker:bus-message-delivery-worker_03: started
bus-message-delivery-worker:bus-message-delivery-worker_04: started
bus-message-delivery-worker:bus-message-delivery-worker_05: started
bus-message-delivery-worker:bus-message-delivery-worker_06: started
bus-message-delivery-worker:bus-message-delivery-worker_07: started
```

Настройка доступа к репозиторию (SSH-ключи)

Для доступа к tandem.gitlab.yandexcloud.net требуется настройка SSH-ключа.

Шаг 1. Генерация ключа

```
ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
# Сохраните в ~/.ssh/id_rsa (по умолчанию)
# Пароль можно оставить пустым (но нежелательно в production)
```

Шаг 2. Добавление в ssh-agent

```
eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_rsa
```

Шаг 3. Передача публичного ключа

Отправьте содержимое ~/.ssh/id_rsa.pub команде TANDEM.

После добавления ключа проверьте доступ:

```
ssh -T git@tandem.gitlab.yandexcloud.net
# Успешный ответ: "Welcome to GitLab, @username!"
```

Важно:

- Не используйте root для генерации и работы с ключами.
- Не меняйте имя файла ключа без явной необходимости.
- Composer должен запускаться от пользователя, которому принадлежит ключ.

Заключение

Соблюдение требований, регулярный мониторинг и плановое обновление — основа стабильной, безопасной и отказоустойчивой работы системы «Личный кабинет».

Помните: мы стараемся сделать каждый релиз максимально надёжным, но окончательная ответственность за эксплуатацию лежит на Заказчике. Ваша внимательность, подготовка и обратная связь — залог долгосрочного успеха.