**Инструкция по работе с web-сервисами филиала**

**“Коммунтеплосбыт” ГУП “ЖКХ РС(Я)”**

Родин В.Д. июль 2020

**Оглавление**

[**Настройка среды разработки**](#_ndx3i3nxsbi1) **3**

[**Основной сервер**](#_l2nehbkkw0q4) **4**

[**Дополнительный сервер**](#_14o0roqh60d5) **7**

[**Инструкция по настройке Ubuntu Server 18**](#_ct5klaumz4em) **7**

[**WEB-серверы**](#_1sr0mfla6i6y) **8**

[**Обслуживание сайта**](#_30j0zll) **9**

[**Функционал платежей**](#_x8vp1s2nwixl) **10**

[**Фискализация**](#_9ophwk4gzzai) **12**

[**Обслуживание сервисов**](#_gp65vgmyam39) **13**

[**Наработки ГИС ЖКХ**](#_b6icl43y6t9o) **14**

[**Краткая инструкция по работе с GIT**](#_8oh4wjvec3we) **16**

# Настройка среды разработки

Для удобства можно использовать готовые сборки серверов (**XAMPP** и подобные). В качестве среды разработки предпочтительно использовать **WebStorm**, **PhpStorm**, но как бесплатный аналог можно использовать **VSCode**.

Для работы с командной строкой Linux можно использовать интегрированные в среды разработки терминалы, либо установить **PuTTY**.

Для работы с текущими исходниками необходимо установить Composer, NodeJS (для NPM).

Для поддержки PHP MSSQL необходимо скачать драйвер с оф. сайта microsoft, затем интегрировать файл в php и добавить его в конфигурацию (раздел extension).

В исходных файлах нет папки uploads проекта **jkhsakha.ru**, т.к. её актуальный объем очень большой. Можно заархивировать на стороне сервера и скачать архив после на локальный компьютер.

*Для проекта BIGAPI (данный проект должен был в будущем выполнять функции главного сервера всех проектов) подразумевалось использовать стеки технологий NodeJS, SocketIO, PHP, Redis, Memcache, MongoDB, MSSQL, LEMP.*

*На текущий этап разработаны тестовые функции для получения данных. Для тестов была создана БД general, загруженная в MYSQL. Для хранения сессий использовали Redis.*

**Ссылка на все исходники проектов (в будущем ссылка может устареть):**

[**https://drive.google.com/file/d/1lQV\_YwiX9dvLQ3XhE6VSwJ4VPu29E7FK/view?usp=sharing**](https://drive.google.com/file/d/1lQV_YwiX9dvLQ3XhE6VSwJ4VPu29E7FK/view?usp=sharing)

# 

# Основной сервер

Основной сервер. ОС – ubuntu 16 x32. Подключение по SSH:

**77.242.1.188 -p 62100**

Для подключения к серверу в Windows, желательно использовать ПО PuTTY.

FTP работает по тому же принципу с указанными выше портами.  
Для каждого сайта имеется свои ftp-юзеры. Можно подключаться по локальному IP-адресу (на тек. момент - 172.16.2.150), а порт указать 21 (**подключение по FTP пассивное!**)

Порты открыты в диапазоне от 62100 до 62500 (насчет доступа портов обращаться к сисадминам ГУП ЖКХ).

Список FTP-пользователей (они ограничены в рамках папок проектов):

**Главный сайт - ftpcli / 159357**

**Сервисы - servicesftp / services159357**

**ГИС (ингтергационные наработки) - gisftp / …**

**ВЕБКВ (неакт.) - webkvftp / …**

**Внутренний сайт организации - coreftp / ...**

Важный момент для работы сайта/сервисов – для поддержки MSSQL в системе linux используется связка FreeTDS+ODBC, а при подключении в PHP использую PDO (указываю наименование конфигурации ODBC – MSSQL/MSSQL385 (файл во всех проектах **/core/dbc.php**),

Конфиг ODBC находится в **/etc/odbc.ini**

Отправка почты с сервера. За отправку почты отвечает **SSMTP** файлы конфигурации можно найти в **/etc/ssmtp**

Почта для email-рассылки

**jkhsakha@gmail.com**

**qaZ159357**

На сервере за рассылку отвечает ssmtp.

Если по каким-то причинам рассылка перестала работать, то проверить сообщения от google на почте (возможно заблокировали) и перейти по ссылке -

[*https://myaccount.google.com/lesssecureapps*](https://myaccount.google.com/lesssecureapps)

(должно быть выбрано- Небезопасные приложения разрешены)

**Let’s Encrypt**

Бесплатный SSL-сертификат, если необходимо добавить поддомен, то надо сначала удалить все данные об let’s encrypt и certbot, а затем переустановить, согласно инструкции.

<https://www.digitalocean.com/community/tutorials/nginx-let-s-encrypt-ubuntu-18-04-ru>

**sudo certbot --nginx \**

**-d jkhsakha.ru \**

**-d api.jkhsakha.ru \**

**-d core.jkhsakha.ru \**

**-d desk.jkhsakha.ru \**

**-d gis.jkhsakha.ru \**

**-d git.jkhsakha.ru \**

**-d services.jkhsakha.ru \**

**-d webkv.jkhsakha.ru**

Планировщик задач CRON

Все задачи, запускаемые автоматически, указаны в crontab, для чтения и редактирования выполнить –

**sudo crontab -e**

В кроне запускаются обычные http-запросы, так как все сервисы написаны на php.

Для работы с исходными файлами проекта на сервере желательно иметь GIT, Composer, NodeJS.

Для подгрузки актуальных исходников из GIT, выполните -

**git pull**

Обратите внимание на файл **.gitignore**, так как в нем не указаны папки uploads, thumbs и некоторые файлы.

В конфиге composer можно увидеть, какие пакеты запрашиваются для работы проекта. Если вы добавили новый пакет, то не нужно загружать его в git, так как в git обновится конфигурационный файл composer, а выполнив на сервере команду подгрузки последних исходников с GIT, выполните в корневой папке проекта (где находится composer.json)

**composer update**

В папке assets было принято решение хранить подключаемые файлы стилей/js, а в папке node\_modules хранить js/css библиотеки, загружаемые из репозитория NPM. Конфигурационный файл npm-а находится в **/assets/package.json/.**

Затем, если необходимо загрузить обновленные/новые пакеты в проект, выполните в папке assets команду

**npm ci**

**Пожалуйста, обратите внимание на то, что крайне не рекомендовано загружать подключаемые библиотеки в GIT, так как это будет лишняя нагрузка на сервер и сам GIT, ибо готовые библиотеки имеют множество ненужных файлов, помимо этого, сложно будет работать с версиями библиотек и следить за их актуальностью.**

# Дополнительный сервер

ОС - Ubuntu 18 x64, подключение по SSH

**172.16.2.222 -p 22**

На нем расположены Docker-образы (система трекинга задач **desk.jkhsakha.ru**), а также GIT (GITLAB opensource). Данные проекты расположены в папке **/var/opt.** Помимо указанных выше проектов, там находятся тестовые сборки REDMINE (аналог openproject), а также есть сборка OPENGKH - готовая сборка для подписания и отправки запросов в ГИС ЖКХ. На текущий момент неизвестно, будет ли работать эта сборка, но, полагаю, что можно попробовать запустить её. О ГИС ЖКХ будет написано в другом разделе.

docker run -d -h gisgkh-soap-php -v '/var/www/opengkh/gisgkh-yii2-example:/example' -p '127.0.0.1:8080:80' --name=gisgkh-soap-php opengkh/gisgkh-soap-php:local`

HTTP-сервер- NGINX, он проксирует запросы к локальным адресам с портами. Описание в разделе ниже.

# Инструкция по настройке Ubuntu Server 18

Ниже приведены ссылки и код для полной настройки сервера.

**NGINX**

$ apt update

$ apt upgrade

$ apt install nginx

**MYSQL**

$ sudo apt update

$ sudo apt install mysql-server

$ sudo mysql\_secure\_installation

Порядок ответов:

N , Y , Y , Y , Y

Добавьте пользователя, который будет иметь доступ, как ROOT:

mysql> CREATE USER 'remote'@'localhost' IDENTIFIED BY '1155Srv05';

mysql> GRANT ALL PRIVILEGES ON \*.\* TO 'remote'@'localhost'

-> WITH GRANT OPTION;

mysql> CREATE USER 'remote'@'%' IDENTIFIED BY '1155Srv05';

mysql> GRANT ALL PRIVILEGES ON \*.\* TO 'remote'@'%'

-> WITH GRANT OPTION;

Для доступа из глобальной сети, отредактируйте конфигурационный файл mysql, и закомментируйте строку ( возможно, /etc/mysql/mysql.conf.d/mysqld.cnf ):

*#bind-address 127.0.0.1*

Перезагрузите сервис

**DOCKER**

Установите docker, согласно данной инструкции:

https://www.digitalocean.com/community/tutorials/docker-ubuntu-18-04-1-ru

После - установите docker-compose

apt install docker-compose

# WEB-серверы

На дополнительном сервере (где находится GIT), установлен WEB-сервер NGINX, он проксирует запросы к локальным адресам с портами. Описание в разделе ниже.

Конфигурационные файлы проектов находятся в **/etc/nginx/sites-enabled**, в которых указаны внутренние редиректы (переадресации).

Также прописаны хосты в **/etc/hosts**, на текущий момент хосты выглядят так:

**127.0.0.1 gitlab.local**

**127.0.0.1 desk.jkhsakha.ru**

**127.0.0.1 git.jkhsakha.ru**

**127.0.0.1 gis.jkhsakha.ru**

На основном сервере установлен NGINX (он основной сервер, он также проксирует запросы на дополнительный сервер). Под NGINX стоит Apache, который управляет всеми веб-сайтами, расположенными на основном сервере. Apache на данном сервере стоял изначально, а NGINX добавился к нему, как надстройка лишь в 2020 году, в тот момент, когда появился доп. сервер.

Конфигурационные файлы находятся в тех-же местах, что и на доп. сервере. Файлы конфигурации Apache находятся в **/etc/apache2**.

# Обслуживание сайта

Стек технологий – PHP 7.1>, JQUERY, CSS, HTML. Движок самописный, по типу MVC, frontend на bootstrap 3 (некоторые страницы начал переводить на 4 версию – проверка ЛС и задолженности на сайте).

Для добавления новых страниц, добавьте сначала путь в роутере (**/core/router.php**), затем создайте контроллер, либо функцию в существующем контроллере (**/ctrl/main.ctrl.php**). А модели находятся в папке **class**. Все вьюшки находятся в папках **/views** , **/pages**. На текущий момент не успели перевести файл **/view/page.php** в нормальный вид.

Иногда, пресс-центр может попросить вас добавить новые пункты меню, либо добавить запись в модальное окно (например, добавить год в список), то, на примере **Раскрытие информации/Тарифы** вам необходимо добавить сначала статическую страницу через админку, взять её ID из списка, добавить пункт в **/views/modals.php** и загрузить файл **modals** на сервер и в GIT.

БД **website** находится на сервере **4.58**.

Функции для мобильных приложений писались еще до перехода на MVC-иерархию, поэтому пути в router-е указаны, как конечные файлы, однако, они вызывают функции внутри контроллера.

На текущий момент имеется приложение для Android (его можно найти в play market). У приложения общая база пользователей и данных с сайтом.

При обращении пользователей с проблемами входа / регистрации - запросить код лс, вызвать хранимую процедуру **website..usp\_kvit** код л/с , 1 (юзер);

Создать тестовый аккаунт через форму регистрации (**test00** / **test00**), добавить л/с туда с данными, отображаемыми в хранимой процедуре.

**Бывают проблемы с полем FIO (могут быть указаны дополнительные данные, эти данные надо ввести в точном соотношении в форме ФИО на сайте)**

**!!! не диктовать и не говорить данные клиентам, просто указывать на то, что они вводят неправильно, а точные данные есть в их ЕПД**

# Функционал платежей

Платежи на сайте, личном кабинете и моб. приложении создаются и обрабатываются в модели **Payment** (**class/payment.php**).

Хранятся платежи в БД **acquiring\_sber**, таблица **reg**.

Структура таблицы **reg** -

**id** - ID записи

**account** - номер Л/С

**amount** - сумма

**id\_jkh** - ID, сгенерированный на стороне нашего сервера

**id\_sber** - GUID, выданный сбербанком

**date\_send** - дата создания платежа

**date\_response** - дата ответа от сбербанка

**result** - результат оплаты со сбербанка (0 - успешно, 1,2 - ошибка)

**approval** - уникальное числовое значение платежа

**up** - проведение успешного завершенного платежа специалистами **ОЦНКП** (у них есть собственное ПО для проведения платежей) (bool 0/1)

**mobile** - bool 0/1, с сайта/приложения

**receipt\_status** - данные по фискализации

**fn\_number** - данные по фискализации

**fiscal\_document\_attribute** - данные по фискализации

**email** - куда будут отправляться чеки (тк не все пользуются эл. почтой, мы подставили стандартное значения example@example.com)

**checked** - проверен ли статус по фискализации

**uuid** - uuid чека

**ecr\_registration\_number** - данные по фискализации

**fiscal\_document\_number** - данные по фискализации

**fiscal\_receipt\_number** - данные по фискализации

**shift\_number** - номер смены (данные по фискализации)

В ПО ОЦНКП, по которому они проводят оплату попадают платежи с логикой result = 0 , approval != ‘’ , approval != ‘0’, approval != ‘000000’. Иногда, банк возвращает такие значения при ошибочных платежах.

В редких случаях бывает так, что в ПО ОЦНКП не попадают некоторые платежи. Им известны суммы, примерные даты, по этим данным платеж можно найти вручную в ЛК Эквайринга сбербанка. Подробнее ниже:

Проверка эквайринга (онлайн-платежей),

доступность сервиса - инициализация платежа на сайте (**jksakha.ru/pay** , тестовые данные - 2002001001 , 1 рубль)

проверка занесения платежа в бд (**select \* from acquiring\_sber..reg order by id desc**)

b) поиск платежей ( если есть обращения от ОЦНКП )

- сначала найти платеж в базе ( **acquiring\_sber..reg** ) , по похожим суммам и датам , просмотр статуса платежа

RESULT 0 - платеж пройден,

2 - платеж не пройден

если **RESULT 0, то APPROVAL != 0 , 000000 , '',** в ином случае - платеж совершен успешно, так же смотри на поля fn fp ( данные ОФД ) , если они пусты, то платеж не прошел (бывают задержки обновления этих данных)

- если платеж не найден - ищем в системе сбербанка - [*https://securepayments.sberbank.ru/mportal/#*](https://securepayments.sberbank.ru/mportal/#)

- если платеж найден в системе сбербанка, но не найден у нас - обсудить решение, возможно сделать возврат платежа.

Помимо платежей на сайте/лк, есть платежи, загружаемые бухгалтерией со сбербанк онлайн для их дальнейшей фискализации. О фискализации написано ниже.

# Фискализация

Очень важный раздел. За фискализацией необходимо периодически следить, в этих целях была разработана автоматическая отправка ежедневных отчетов (в **services**) по платежам.

Фискализация работает следующим образом:

1. Создание платежа пользователем;
2. Ввод реквизитов и завершение платежа;
3. Сбербанк (если это оплата на сайте/лк) / Сервис обработки платежей СберОнлайн отправляют запрос в **АТОЛ** (фискальный аппарат находится именно там);
4. **АТОЛ** регистрирует платеж и отправляет его в ОФД (**ПлатформаОФД**);
5. Возвращается ответ с результатом фискализации.

Платежи, приходящие из **СберОнлайн** хранятся в БД **acquiring\_sber**, в таблице **kvit.**

Структура таблицы **kvit**-

**no\_kv** - номер квитанции (уникальное число)

**postfix** - вспомогательный параметр (если по no\_kv фискализация прошла с ошибкой и невозможно фискализировать заново, то добавьте в postfix запись типа -220 (к примеру), тогда для АТОЛ это будет новая операция. **Не перезаписывать postfix неглядя! тк вы можете потерять платеж!**)

**kod\_adres** - номер Л/С

**summa** - сумма

**peni** - пеня

**data\_opl** - дата оплаты

**data\_gash** - дата проведения платежа

**kod\_operator** - ID пользователя (из БД SPR)

**kod\_kassa** - ID кассы (из БД SPR)

**oper** - тип операции (0 - оплата, 1 - возврат)

**email** - куда будут отправляться чеки

**uuid** - GUID платежа в системе АТОЛ

**status** - статус операции (3 типа: **fail** - ошибка, **wait** - ожидание результата, **done** - завершено)

**receipt\_status** - данные по фискализации

**fn\_number** - данные по фискализации

**fiscal\_document\_attribute** - данные по фискализации

**uuid** - uuid чека

**ecr\_registration\_number** - данные по фискализации

**fiscal\_document\_number** - данные по фискализации

**fiscal\_receipt\_number** - данные по фискализации

**shift\_number** - номер смены (данные по фискализации)

**error\_text** - если произошла какая-то ошибка в обработке платежа - она запишется сюда в виде текста

АТОЛ предоставляет в аренду ККТ Онлайн, через который мы фискализируем платежи. Оплата за их услуги происходит ежегодно, а договор с ПлатформаОФД в рамках сотрудничества с АТОЛ.

Важно следить за уведомлениями от АТОЛ, так как могут произойти такие случаи:

1. Касса будет неактивна (тех работы и пр.);
2. У кассы истечет ФН (фискальный накопитель), который необходимо ***регистрировать в ФНС, а старый ФН снимать с регистрации (!!!);***
3. Другие случаи.

***Важно не упускать моменты, в которые касса перестает функционировать, а платежи принимаются до сих пор, платежи необходимо отключить, можно поставить заглушку в исходниках в Payment.php, устранить проблему как можно скорее.***

Фискализация после оплаты происходит не сразу, поэтому результат фискализации приходит позже. Сервис по обработке платежей будет описан далее.

# Обслуживание сервисов

БОльшая часть сервисов находится в проекте **services**, однако, частично сервисы есть и в **jkhsakha.ru**.

Актуальные сервисы можно увидеть в CRON.

В **jkhsakha.ru** осталась обработка платежей из сайта (сервис в автоматическом режиме с заданной периодичностью в cron-е обходит платежи, которые не были завершены, чтобы получить по ним конечный статус, которые завершены, но код approval не был подгружен и завершенные платежи, по которым нет статуса фискализации.

Так выглядит вызов в CRON- (Раз в 5 минут обход платежей):

**\*/5 \* \* \* \* curl** [**http://jkhsakha.ru/sbercheck.php**](http://jkhsakha.ru/sbercheck.php)

В services находятся сервисы:

1. По парсингу погоды из Yahoo (ранее использовались другие поставщики данных);
2. Парсинг ключевой ставки ЦБ РФ;
3. Запросы по получению кодов ФИАС по объектам, в которых данные коды отстутствуют;
4. Автоматическая фискализация платежей, загруженных из СберОнлайн;
5. Автоматическая отправка на эл. почту ежедневного отчета по платежам;
6. Функционал генерирования ЕПД в формате PDF (вызывается из личного кабинета и приложения Сбербанк.Онлайн).

**Для формирования ЕПД в формате PDF используется библиотека wkhtmltox\_0.12.5-1.xenial\_i386. Версия должна быть именно 0.12.5-1, при работе с другими версиями может нарушиться структура документа!**

# Наработки ГИС ЖКХ

При разработке прямой интеграции с ГИС ЖКХ мы пытались реализовать 3 способа:

1. Подписание SOAP-запроса через доп.сервер на windows, так как сервер вызывал исполняемый файл, подписывающий запрос и возвращал ответ (вариант был самым подходящим, но перестал работать в момент, когда организации выдали сертификат с ГОСТ-2012 (ранее был 2004));
2. Сборка CURL с поддержкой GOST на Linux - не вышло;
3. Использовать docker-образ примера интеграции OPENGKH (до конца неясно, рабочий ли это вариант, надо попробовать).

Данные тестовой среды -

Доступ к СИТ Basic Авторизация

sit / xw{p&&Ee3b9r8?amJv\*]

Доступ в тестовый ЛК -

OOOSHKHRST1 / 123qweASD

1. Использование стороннего ПО

1.1. Обязательно использование КриптоПРО CSP от 4 версии

1.2. МагПро Криптотуннель - приобретенное ПО для системы LINUX (дистрибутив имеется)

1.3. Ежегодная покупка ПО (3990 рублей) P12FROMGOST от Lissisoft для экспорта ключа сертификата

1.4. Ежегодная заявка на демо-версию ПО МагПро openssl (для экспорта сертификата)

2. Работа с сертификатами

2.1. Общая информация

Сертификат должен быть КС2 уровня с возможностью экспорта закрытого ключа, обычно, сертификат имеет срок действия 1 год, при работе с новыми сертификатами, необходимо проводить операции по экспорту ключей и т.д. заново.

2.2. Выполните импорт сертификата из ключа на локальный компьютер

2.3. Выполните экспорт сертификата в файл без ключа в формате DER

2.4. Выполните экспорт сертификата с ключом через утилиту P12FROMGOST

2.5. Выполните в openssl (МагПро - важно(!) т.к. с поддержкой gost) следующие команды:

x509 -inform der -in cert.cer -out cert.pem

x509 -outform der -in cert.pem -out cert.crt

pkcs12 -in <filename>.pfx –nodes -out <filename>.key

3. Импорт сертификатов

3.1. Общая информация

При загрузке новых сертификатов важно загружать сертификат как на рабочую среду, так и на актуальную тестовую (в данном случае - СИТ02).

3.2. Загрузка сертификата .cer на ресурс dom.gosuslugi - загрузите в /operator/systems и в сит02.

3.3. Добавление сертификата в криптотуннель

3.3.1. Загрузите сертификат формата .pem и ключ формата .key в папку /keys web-сервиса ГИС ЖКХ (на сервере);

3.3.2. Откройте файл /etc/stunnel-gost-client/stunnel.conf, поменяйте названия файлов на загруженные файлы сертификата в конце документа (переменные cert,key), сохраните файл и перезапустите криптотуннель, выполнив команду -

sudo -H /etc/init.d/stunnel-gost-client restart

*Актуальную документацию по работе с ГИС ЖКХ можно найти в dom.gosuslugi.ru. В разделе* ***Регламенты и инструкции - Взаимодействие с внешними системами - Регламент и форматы информационного взаимодействия внешних информационных систем с ГИС ЖКХ (текущие форматы v.13.1.1.5) (версия может отличаться).***

# Краткая инструкция по работе с GIT

[git.jkhsakha.ru](http://git.jkhsakha.ru)

1. Создание репозитория

На вкладке Projects нажимаем на New project (кнопка в правом углу экрана);

Пишем название проекта (постараться это сделать максимально коротко, на англ. языке);

2. Инициализация GIT в проекте (через командную строку (некоторые IDE позволяют настроить GIT прямо в них же))

Скачайте и установите клиент GIT;

Перейдите в папку с проектом из командной строки;

В корневой папке проекта введите команду - git init;

Заполните файл .gitignore (примеры найдите в других проектах)...

... Здесь важно отметить то, что этот файл отсекает ненужные файлы для загрузки в репозиторий (в репозитории должны быть только рабочие файлы), например, отсечь .exe, .log файлы, папки с архивами и сами архивы;

Затем, согласно настройкам вашего проекта в GIT выполните команды на примере jkhsakha:

**git remote add origin http://git.jkhsakha.ru/viktorr/jkhsakha.git**

**git add .**

**git commit -m "First commit"**

**git push -u origin master**

Пройдите авторизацию в командной строке (её потребует сам GIT).

Остальную информацию по GIT можно найти в интернете :)