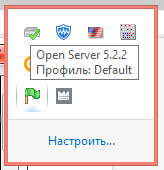
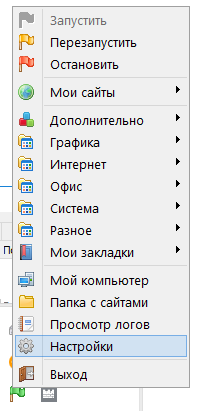
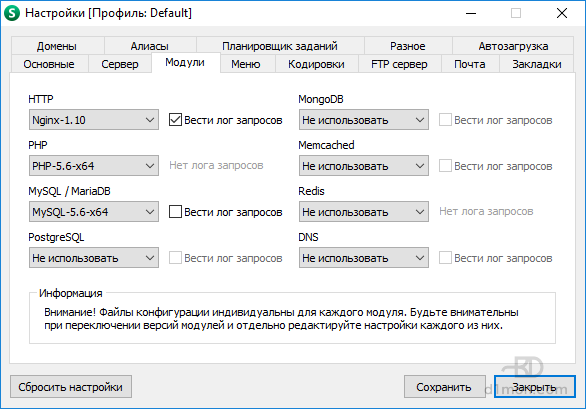
**ЧАСТЬ 1. УСТАНОВКА И НАСТРОЙКА YII 2.0 НА OPEN SERVER**

Перед установкой фреймворка Yii необходимо установать web-сервер **Open Server**. Файлы с установкой расположены здесь: open-server.ru. Рекомендуется использовать Open Server в редакции «premium», которая сразу включает **Git for Windows**, чтобы не устанавливать его отдельно.

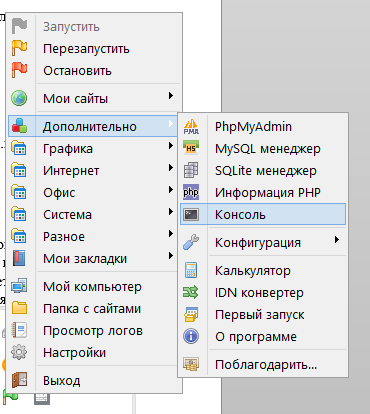
По умолчанию локальный сервер (при стандартной установке) находится в каталоге C:\OpenServer. Сайт на Yii2 будет в каталоге C:\OpenServer\domains\test.yii.

В настройках Open Server выбираем Ngnix 1.10, PHP 5.6-x64, MySQL 5.6-x64. Для перехода в Настройки необходимо выбрать соответствующий пункт на сервере.



Запускаем Open Server от имени администратора, открываем консоль:



Все команды, которые последуют далее, необходимо выполнять именно в консоли Open Server. Для начала убедимся, что Git работает:

git

Если получили в ответ справку, значит всё в порядке. Это важно, т.к. при установке движка Yii2 и его расширений Git используется. Для установки Yii всего использовать Cоmposer. Сначала Composer надо установить в Open Server. Сделать это можно в каталог C:\OpenServer. Если требуется, перейдите в этот каталог:

cd c:\openserver

Для установки выполните поочередно следующие команды:

php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"

php -r "if (hash\_file('SHA384', 'composer-setup.php') === 'e115a8dc7871f15d853148a7fbac7da27d6c0030b848d9b3dc09e2a0388afed865e6a3d6b3c0fad45c48e2b5fc1196ae') { echo 'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.php'); } echo PHP\_EOL;"

php composer-setup.php

php -r "unlink('composer-setup.php');"

Первая скачивает инсталлятор Composer, вторая проверяет его, третья производит установку, последняя удаляет уже не нужный файл. Получаем:

Downloading 1.2.0...

Composer successfully installed to: C:\OpenServer\composer.phar

Если данный вариант установки не работает, можно отдельно скачать и установить Composer. Файл установщика для Windows можно скачать здесь <https://getcomposer.org/doc/00-intro.md#installation-nix>

**Примечание!**

При установке композера из файла необходимо вручную выбрать путь к php, заданному в настройках сервера выше. В нашем варианте это PHP 5.6-x64. При это директория с файлом php.exe будет С:\OpenServer\modules\php\PHP-5.6-x64

Дополнительно для Composer устанавливаем плагин:

composer global require "fxp/composer-asset-plugin:~1.1.1"

Он тоже сообщает об успешном скачивании и т.д. Теперь можно устанавливать сам Yii2. Разработчики Yii2 предлагают устанавливать движок, выбирая модификацию, содержащую по умолчанию один из двух шаблонов — **basic** или **advanced**. Последний больше подходит для традиционных сайтов, имеющих админку. В нём, как бы, два сайта в одном — **frontend** (сайт для всех) и **backend** (сайт-админка). Эти сайты имеют общие и раздельные настройки, общие и раздельные элементы и т.д. Это очень удобно и полезно. Шаблон **basic** — это просто один сайт. Здесь нет понятия frontend и backend. Сайт для всего — общий.

Переходим в каталог сайта и устанавливаем Yii2 с advanced-шаблоном:

cd domains\test.yii

composer create-project yiisoft/yii2-app-advanced ./ 2.0.10

Процесс установки Yii2 занимает порядка 3-5 минут. Скачиваются нужные пакеты, распаковываются, раскладываются в нужные каталоги. Всё происходит автоматически.

В процессе установки потребуется указать OAuth Token для GitHub, если таковой не был указан ранее. Он запоминается в системе и поэтому делается это один раз. Токен открывает доступ для автоматизированной загрузки с GitHub некоторых пакетов, необходимых Yii2.

Если у вас ещё нет аккаунта на GitHub, то регистрируйтесь по адресу https://github.com/. Сервис предоставляет вариант бесплатного использования, выбирайте его. Бесплатно — это публичное размещение исходников. Приватное размещение на GitHub только за деньги. В процессе установки запрос токена происходят в таком виде:

please create a GitHub OAuth token to go over the API rate limit

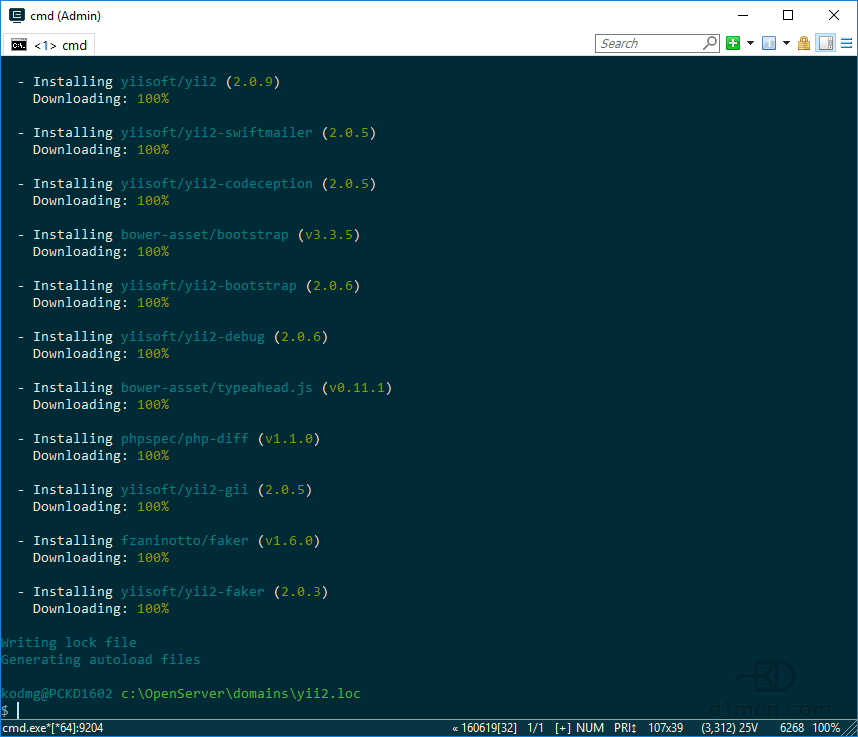
Head to https://github.com/settings/tokens/new?scopes=repo&description=Composer+for+Open+Server

to retrieve a token. It will be stored in "C:/Users/kodmg/AppData/Roaming/Composer/auth.json" for future use by Composer.

Token (hidden):

Token stored successfully.

Т.е. нам даже дают ссылку по которой надо пойти, чтобы создать токен. Сгенерированный токен копируем и вставляем в консоль, его будет не видно (пример выше). Просто вставили, жмём «Enter» и установка идёт дальше. Установка Yii2 завершена:



Перед тем, как начать пользоваться нужно произвести инициализацию, в каталоге сайта выполняем:

init

Выбираем «Development» (разработка), подтверждаем «yes», готово:

$ init

Yii Application Initialization Tool v1.0

Which environment do you want the application to be initialized in?

[0] Development

[1] Production

Your choice [0-1, or "q" to quit] 0

Initialize the application under 'Development' environment? [yes|no] yes

Start initialization ...

generate backend/config/main-local.php

generate backend/config/params-local.php

...

generate yii

generate cookie validation key in backend/config/main-local.php

generate cookie validation key in frontend/config/main-local.php

chmod 0777 backend/runtime

chmod 0777 backend/web/assets

chmod 0777 frontend/runtime

chmod 0777 frontend/web/assets

chmod 0755 yii

chmod 0755 tests/codeception/bin/yii

... initialization completed.

Смысл инициализации состоит в первоначальной подготовке некоторых ключевых файлов, каталогов и настроек сайта. Мы выбираем вариант «Development» и нам дополнительно подключается, например, дебагер, встроенный в Yii2.

Повторно выполнять команду инициализации потом нельзя, она все файлы настроек выставляет в вид по умолчанию. Команда используется именно на начальном этапе, когда ещё ничего нет.

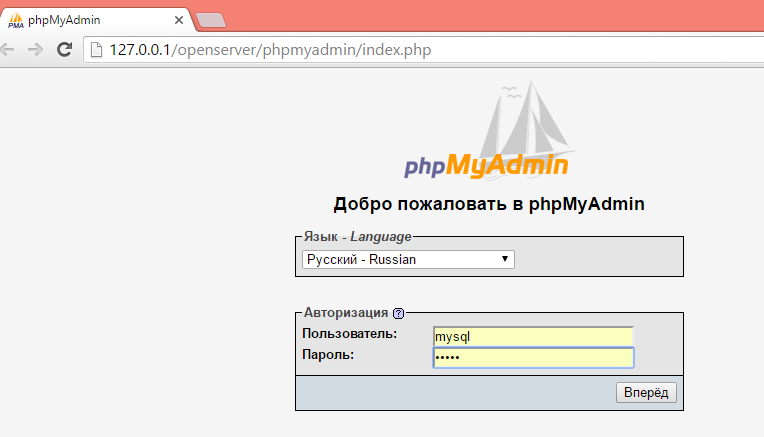
Заходите на сайт, используя ссылку test.yii/frontend/web

Админская часть находится здесь test.yi/backend/web

Чтобы войти в админку, нужно вначале зарегистрироваться на сайте, а для этого нужно создать базу данных.

Через phpMyAdmin создаём базу для сайта. Например «db\_yii2», кодировку при этом выбираем «utf8\_general\_ci».

По умолчанию пароль и логин для входа в phpmyadmin: mysql



Редактируем файл настроек сайта common/config/main-local.php

<?php

return [

'components' => [

'db' => [

'class' => 'yii\db\Connection',

'dsn' => 'mysql:host=localhost;dbname=db\_yii2',

'username' => 'root',

'password' => '',

'charset' => 'utf8',

],

'mailer' => [

'class' => 'yii\swiftmailer\Mailer',

'viewPath' => '@common/mail',

// send all mails to a file by default. You have to set

// 'useFileTransport' to false and configure a transport

// for the mailer to send real emails.

'useFileTransport' => true,

],

],

];

Здесь прописываете имя базы, имя пользователя базы, пароль (если есть).

Теперь в консоли выполняем миграцию:

yii migrate

Будет найден единственный файл миграции, который ещё не применялся, подтверждаем его применение.

$ yii migrate

Yii Migration Tool (based on Yii v2.0.10)

Creating migration history table "migration"...Done.

Total 1 new migration to be applied:

m130524\_201442\_init

Apply the above migration? (yes|no) [no]:yes

\*\*\* applying m130524\_201442\_init

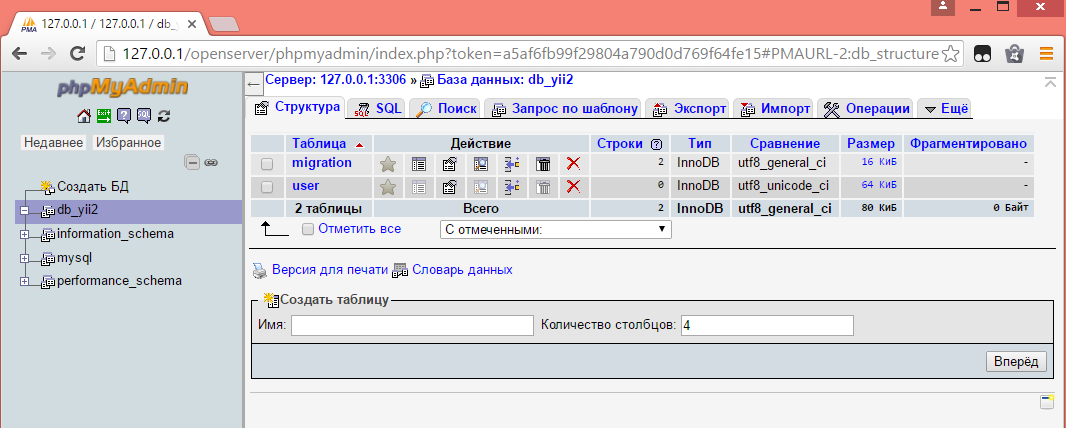
> create table {{%user}} ... done (time: 0.060s)

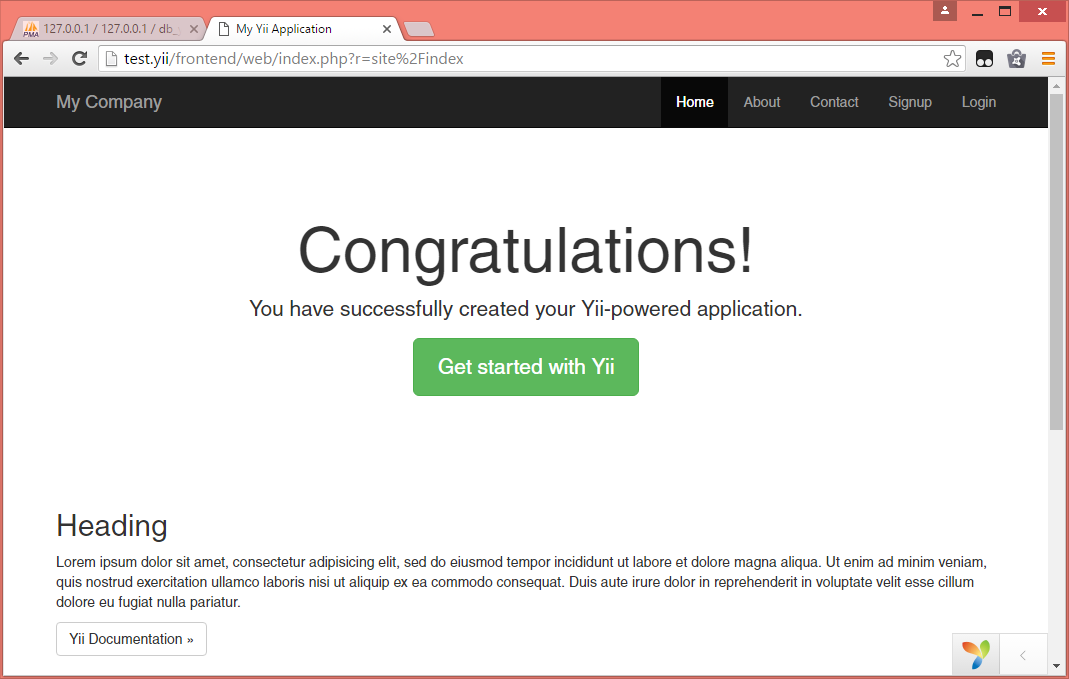
\*\*\* applied m130524\_201442\_init (time: 0.075s)

1 migration was applied.

Migrated up successfully.

В базе появятся две таблицы. Теперь на сайте можно регистрироваться.





Чтобы указанные ссылки привести к виду test.yii и test.yii/backend, соответственно, необходимо настроить Nginx.

В Open Server это делается так. В зависимости от выбранной версии Nginx в модулях в панели настроек, мы берём файл конфигурации из каталога C:\OpenServer\userdata\config\ и копируем в каталог сайта C:\OpenServer\domains\test.yii\.

Для Nginx 1.10 копируем файл **Nginx-1.10\_vhost.conf**

Теперь, всё, что мы настроим в этом файле, будет распространятся на Nginx только для этого сайта. Это очень удобно! Весь сайты могут быть разные, на разных движках. Каждому обычно нужны свои настройки.

Нужно перезапустить Open Server, чтобы настройки сработали.

Вариант конфигурации:

#-----------------------------------------------#

# Начало блока конфигурации хоста

#-----------------------------------------------#

server {

listen %ip%:%httpport%;

listen %ip%:%httpsport% ssl;

server\_name %host% %aliases%;

index index.php;

ssl\_certificate "%sprogdir%/userdata/config/server.crt";

ssl\_certificate\_key "%sprogdir%/userdata/config/server.key";

#add\_header Strict-Transport-Security "max-age=94608000; includeSubDomains; preload";

# if ($request\_method !~\* ^(GET|HEAD|POST)$ ){return 403;}

location ~ /\. {deny all;}

location ^~ /backend {

rewrite ^/backend(.\*)$ /backend/web$1 last;

}

location ^~ /backend/web {

root "%hostdir%";

location ~ \.php$ {

try\_files $uri =404;

# if (!-e $document\_root$document\_uri){return 404;}

fastcgi\_pass backend;

fastcgi\_index index.php;

fastcgi\_buffers 4 64k;

fastcgi\_connect\_timeout 1s;

fastcgi\_ignore\_client\_abort off;

fastcgi\_next\_upstream timeout;

fastcgi\_read\_timeout 5m;

fastcgi\_send\_timeout 5m;

fastcgi\_param CONTENT\_TYPE $content\_type;

fastcgi\_param CONTENT\_LENGTH $content\_length;

fastcgi\_param DOCUMENT\_URI $document\_uri;

fastcgi\_param DOCUMENT\_ROOT $document\_root;

fastcgi\_param GATEWAY\_INTERFACE CGI/1.1;

fastcgi\_param HTTPS $https;

fastcgi\_param QUERY\_STRING $query\_string;

fastcgi\_param REQUEST\_METHOD $request\_method;

fastcgi\_param REQUEST\_URI $request\_uri;

fastcgi\_param REMOTE\_ADDR $remote\_addr;

fastcgi\_param REMOTE\_PORT $remote\_port;

fastcgi\_param SERVER\_ADDR $server\_addr;

fastcgi\_param SERVER\_PORT $server\_port;

fastcgi\_param SERVER\_NAME $host;

fastcgi\_param SERVER\_PROTOCOL $server\_protocol;

fastcgi\_param SERVER\_SOFTWARE nginx;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

fastcgi\_param SCRIPT\_NAME $fastcgi\_script\_name;

fastcgi\_param TMP "%sprogdir%/userdata/temp";

fastcgi\_param TMPDIR "%sprogdir%/userdata/temp";

fastcgi\_param TEMP "%sprogdir%/userdata/temp";

}

}

location / {

root "%hostdir%/frontend/web";

try\_files $uri $uri/ /index.php$is\_args$args;

location ~ \.php$ {

try\_files $uri =404;

# if (!-e $document\_root$document\_uri){return 404;}

fastcgi\_pass backend;

fastcgi\_index index.php;

fastcgi\_buffers 4 64k;

fastcgi\_connect\_timeout 1s;

fastcgi\_ignore\_client\_abort off;

fastcgi\_next\_upstream timeout;

fastcgi\_read\_timeout 5m;

fastcgi\_send\_timeout 5m;

fastcgi\_param CONTENT\_TYPE $content\_type;

fastcgi\_param CONTENT\_LENGTH $content\_length;

fastcgi\_param DOCUMENT\_URI $document\_uri;

fastcgi\_param DOCUMENT\_ROOT $document\_root;

fastcgi\_param GATEWAY\_INTERFACE CGI/1.1;

fastcgi\_param HTTPS $https;

fastcgi\_param QUERY\_STRING $query\_string;

fastcgi\_param REQUEST\_METHOD $request\_method;

fastcgi\_param REQUEST\_URI $request\_uri;

fastcgi\_param REMOTE\_ADDR $remote\_addr;

fastcgi\_param REMOTE\_PORT $remote\_port;

fastcgi\_param SERVER\_ADDR $server\_addr;

fastcgi\_param SERVER\_PORT $server\_port;

fastcgi\_param SERVER\_NAME $host;

fastcgi\_param SERVER\_PROTOCOL $server\_protocol;

fastcgi\_param SERVER\_SOFTWARE nginx;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

fastcgi\_param SCRIPT\_NAME $fastcgi\_script\_name;

fastcgi\_param TMP "%sprogdir%/userdata/temp";

fastcgi\_param TMPDIR "%sprogdir%/userdata/temp";

fastcgi\_param TEMP "%sprogdir%/userdata/temp";

}

}

# Подключение веб-инструментов

#---------------------------------------#

# <Не изменяйте этот блок конфигурации>

location /openserver/ {

root "%sprogdir%/modules/system/html";

index index.php;

%allow%allow all;

allow 127.0.0.0/8;

allow ::1/128;

allow %ips%;

deny all;

location /openserver/server-status {

stub\_status on;

}

location ~ ^/openserver/.\*\.php$ {

root "%sprogdir%/modules/system/html";

fastcgi\_pass backend;

fastcgi\_index index.php;

fastcgi\_buffers 4 64k;

fastcgi\_connect\_timeout 1s;

fastcgi\_ignore\_client\_abort off;

fastcgi\_next\_upstream timeout;

fastcgi\_read\_timeout 5m;

fastcgi\_send\_timeout 5m;

fastcgi\_param CONTENT\_TYPE $content\_type;

fastcgi\_param CONTENT\_LENGTH $content\_length;

fastcgi\_param DOCUMENT\_URI $document\_uri;

fastcgi\_param DOCUMENT\_ROOT $document\_root;

fastcgi\_param GATEWAY\_INTERFACE CGI/1.1;

fastcgi\_param HTTPS $https;

fastcgi\_param QUERY\_STRING $query\_string;

fastcgi\_param REQUEST\_METHOD $request\_method;

fastcgi\_param REQUEST\_URI $request\_uri;

fastcgi\_param REMOTE\_ADDR $remote\_addr;

fastcgi\_param REMOTE\_PORT $remote\_port;

fastcgi\_param SERVER\_ADDR $server\_addr;

fastcgi\_param SERVER\_PORT $server\_port;

fastcgi\_param SERVER\_NAME $host;

fastcgi\_param SERVER\_PROTOCOL $server\_protocol;

fastcgi\_param SERVER\_SOFTWARE nginx;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

fastcgi\_param SCRIPT\_NAME $fastcgi\_script\_name;

fastcgi\_param TMP "%sprogdir%/userdata/temp";

fastcgi\_param TMPDIR "%sprogdir%/userdata/temp";

fastcgi\_param TEMP "%sprogdir%/userdata/temp";

}

}

# <Не изменяйте этот блок конфигурации/>

#---------------------------------------#

}

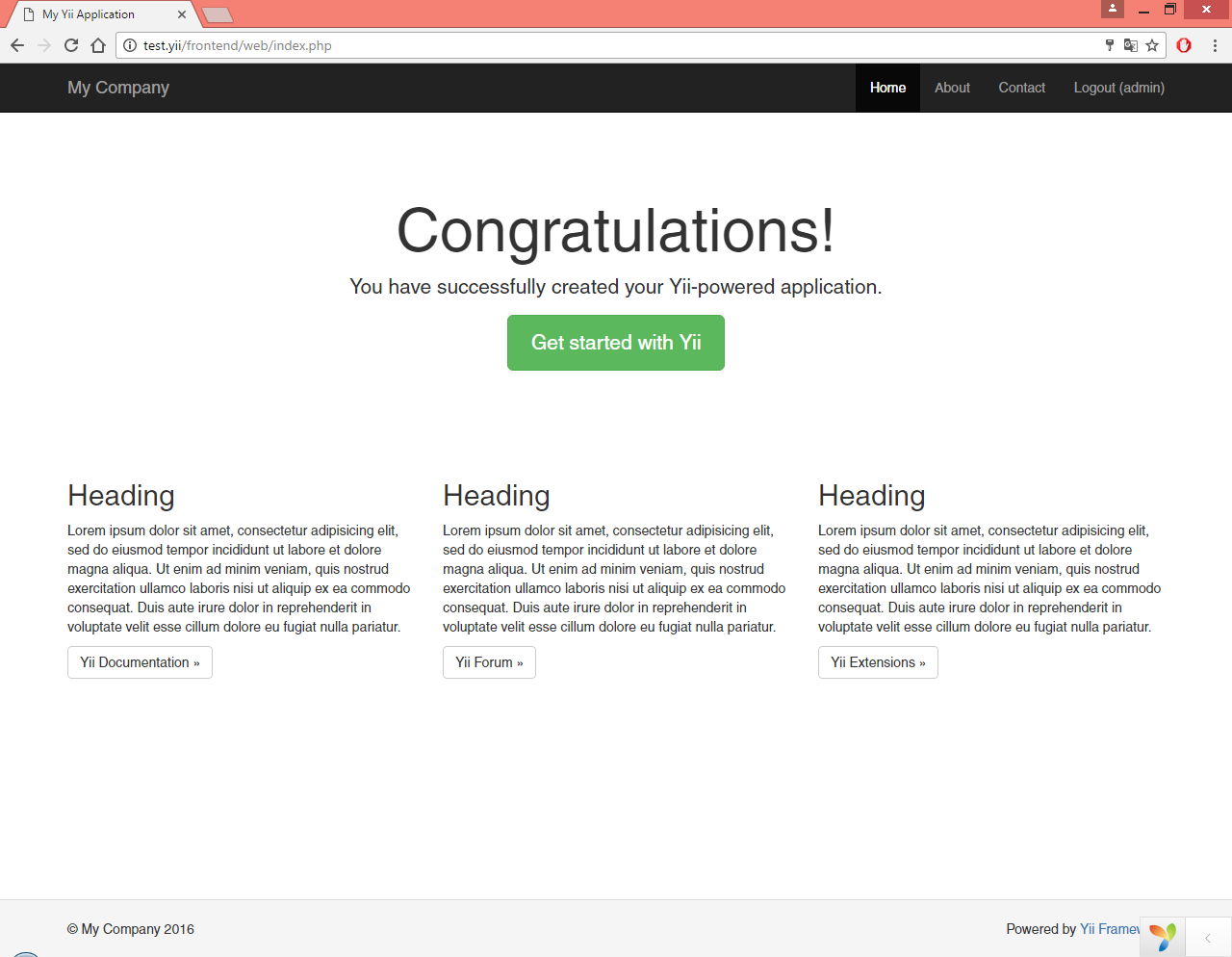
#-----------------------------------------------#

# Конец блока конфигурации хоста

#-----------------------------------------------#

После выполнения данных настроек необходимо еще раз перезапустить сервер.

Далее регистрируем нового пользователя с данными: логин – admin, пароль – admin1. Входим на сайт, используя его регистрационные данные.



Просмотрим содержимое базы данных. Если все было выполнено правильно, в таблице user появится новая запись.

