WinnRepo

WinnRepo will be a Software as a Service (SaaS) to help provide a better gateway for software developers of **Premium** type of plugins and themes, namely for WordPress. Where we’ll allow them access per a monthly fee. Where it’ll also be way more affordable with better functions than what you see here: <http://wp-updates.com/>

We’re basically going to build a better wp-updates.com by creating it under the WinnComm, LLC brand and support will be handled by [www.WinnComm.net](http://www.winncomm.net) employees.

The concept is to use wp-updates.com as an example of what we’re building and clone their idea but we’re using [larabel framework](https://laravel.com/) for this, as an easy to use framework to update on the fly. :)

cPanel login:

URL: <https://s1.winncomm.com:2083/>

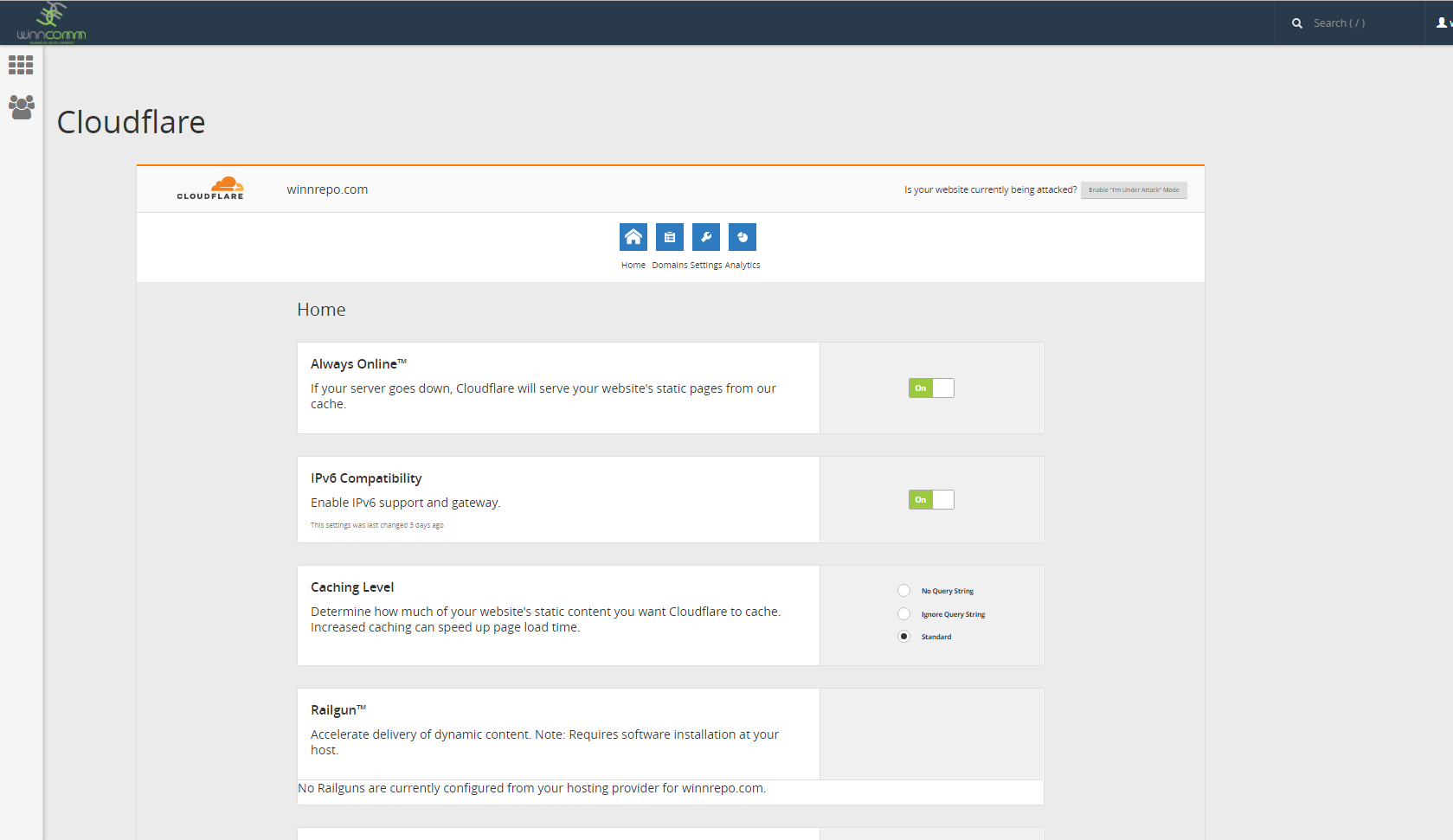
Username: winnrepo

Password: ;t[6m9oe&IRA

Within there, you can make your own FTP setup, or upload directly within cPanel.

The site is brand new, and ready for testing, I advise to setup a subdomain for staging. Ask Shapon what I mean by this, it will be for testing before going live. You can set this up within cPanel under the Subdomains section. I’ll have to handle the changes on CloudFlare if needed once setup. I personally will more than likely have it setup before work starts.

As you can see this is already set up through CloudFlare API now within the server to handle DNS and whatever cache from within cPanel:



This is a joint effort project, where WinnComm will handle all of the marketing, getting the support of the service for the clients and etc… Then Shapon and his team will be the development team on this project, and we’ll be splitting everything 50/50 on payment after monthly expenses for the servers and the usage on (WinnComm) servers. However, the cost at first will not be expensive because I’m going to try and route as much as possible through CloudFlare.

Site Structure

Don’t worry about everything looking pretty but we’ll need to get the following areas ready. We can make just a landing page only for the front-end.

## User Roles

We need a admin section to control the users, their abilities and etc along with roles for them. Meaning we’ll have a structure as:

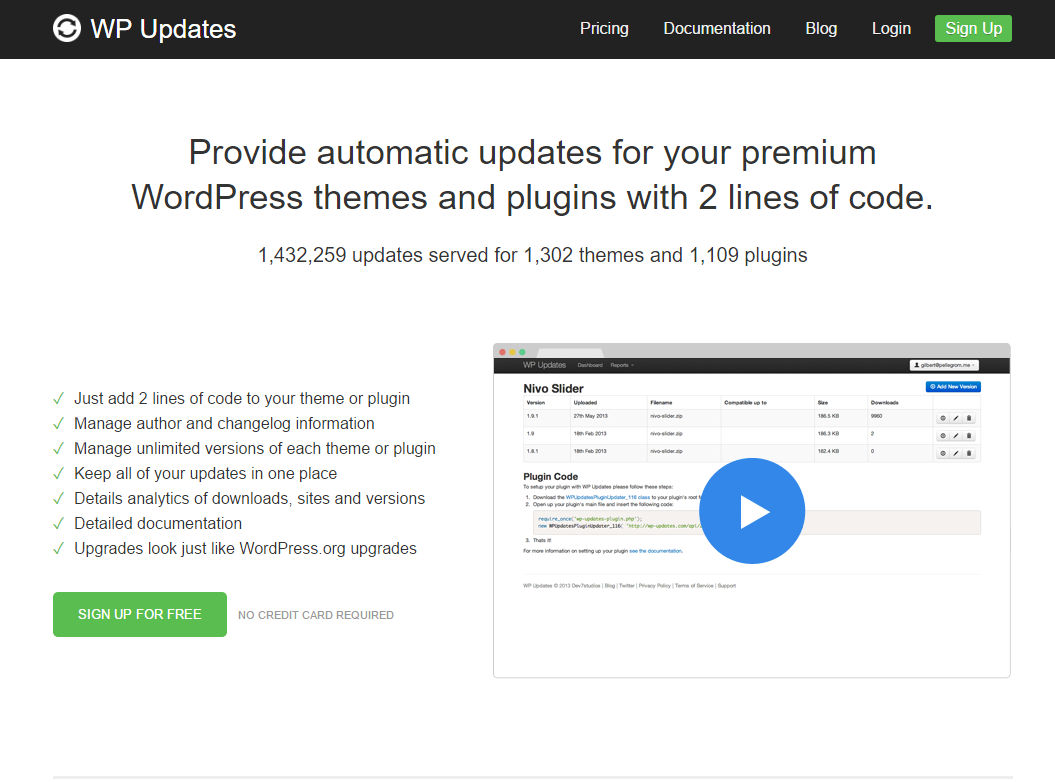
Admin

Paid User

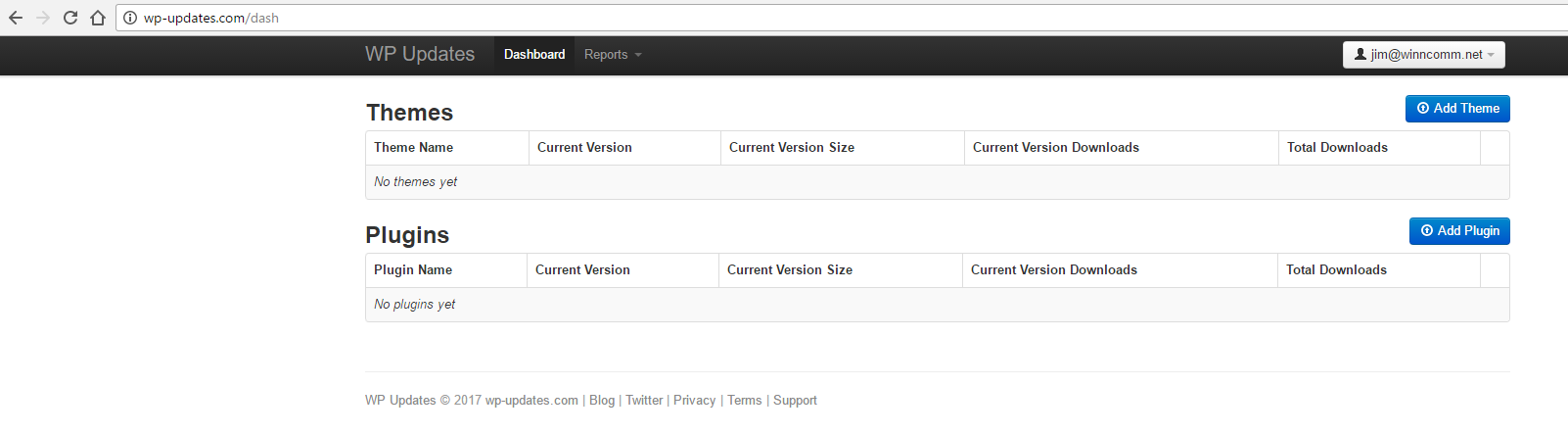
Non-Paid User (Cancelled account)

Test User (Development tests)

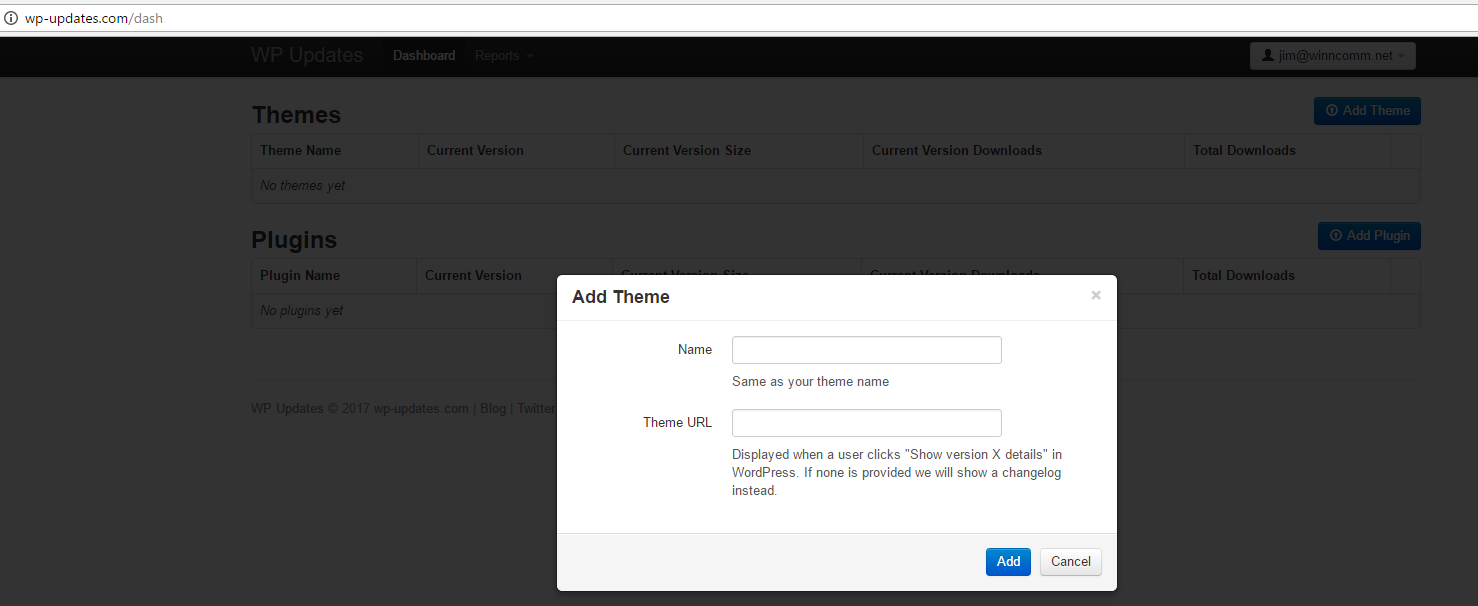
## Clone Concept and Screenshots (Front End)



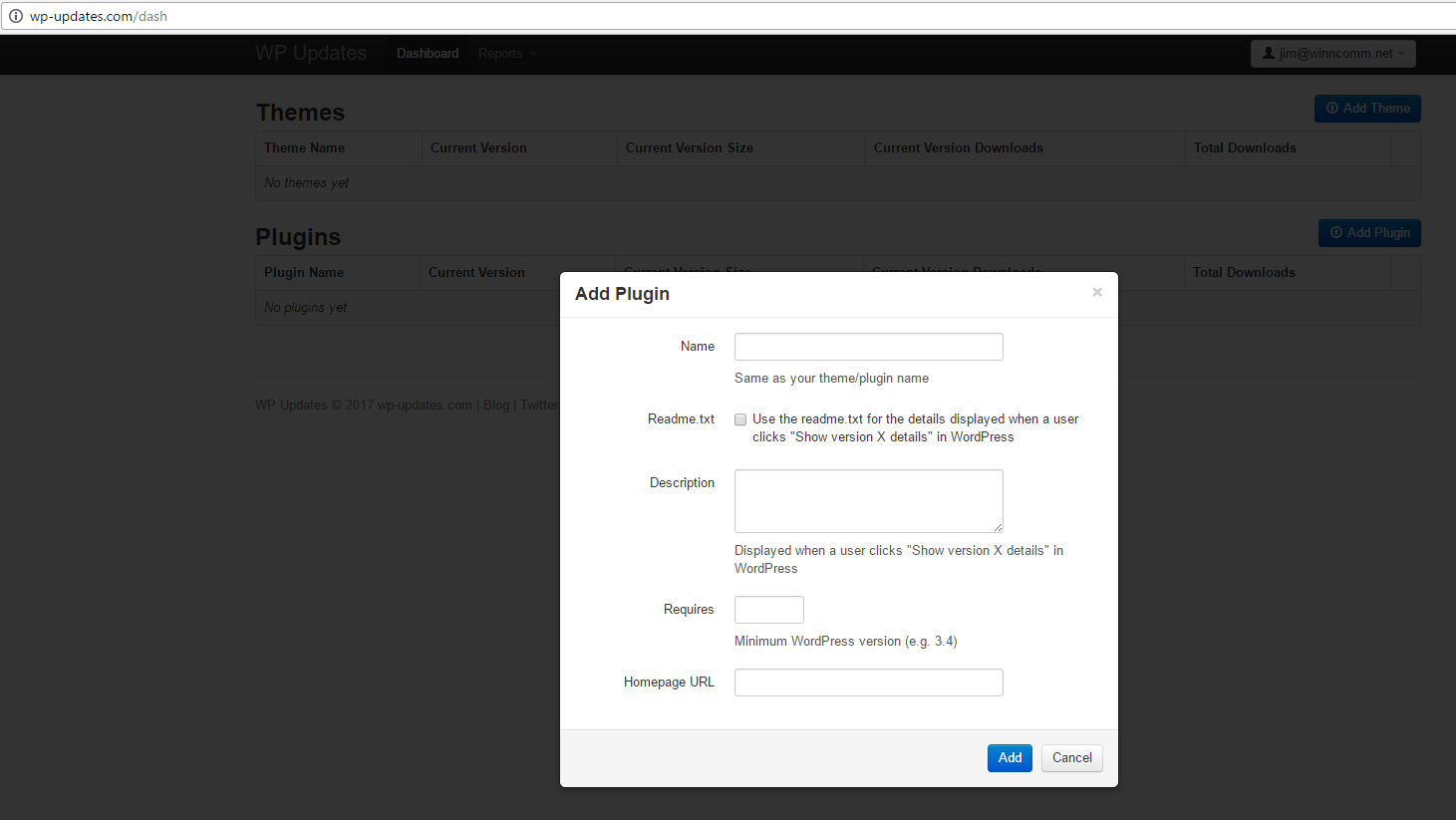
Their Dashboard:



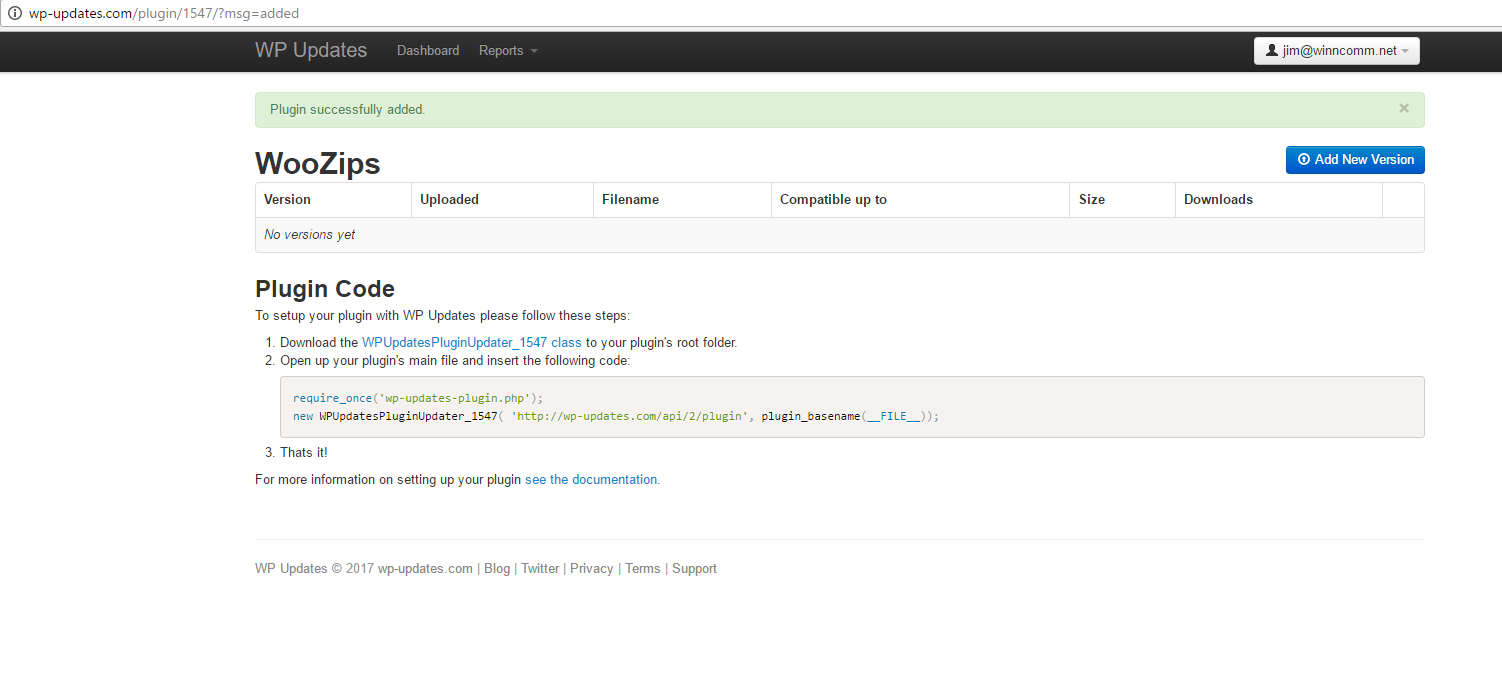
Their Add Theme:



Their Add Plugin:



Added a plugin, and the following shows:



Which states and live links to see the class file.:

## **Plugin Code**

To setup your plugin with WP Updates please follow these steps:

1. Download the [WPUpdatesPluginUpdater\_1547 class](http://wp-updates.com/code/plugin/1547) to your plugin's root folder.
2. Open up your plugin's main file and insert the following code:
3. require\_once('wp-updates-plugin.php');  
   new WPUpdatesPluginUpdater\_1547( 'http://wp-updates.com/api/2/plugin', plugin\_basename(\_\_FILE\_\_));
4. That's it!

For more information on setting up your plugin [see the documentation](http://wp-updates.com/documentation#plugin).

Class file contains:

<?php

/\*

WPUpdates Plugin Updater Class

http://wp-updates.com

v2.0

Example Usage:

require\_once('wp-updates-plugin.php');

new WPUpdatesPluginUpdater\_1547( 'http://wp-updates.com/api/2/plugin', plugin\_basename(\_\_FILE\_\_) );

\*/

if( !class\_exists('WPUpdatesPluginUpdater\_1547') ) {

class WPUpdatesPluginUpdater\_1547 {

var $api\_url;

var $plugin\_id = 1547;

var $plugin\_path;

var $plugin\_slug;

var $license\_key;

function \_\_construct( $api\_url, $plugin\_path, $license\_key = null ) {

$this->api\_url = $api\_url;

$this->plugin\_path = $plugin\_path;

$this->license\_key = $license\_key;

if(strstr($plugin\_path, '/')) list ($t1, $t2) = explode('/', $plugin\_path);

else $t2 = $plugin\_path;

$this->plugin\_slug = str\_replace('.php', '', $t2);

add\_filter( 'pre\_set\_site\_transient\_update\_plugins', array(&$this, 'check\_for\_update') );

add\_filter( 'plugins\_api', array(&$this, 'plugin\_api\_call'), 10, 3 );

// This is for testing only!

//set\_site\_transient( 'update\_plugins', null );

// Show which variables are being requested when query plugin API

//add\_filter( 'plugins\_api\_result', array(&$this, 'debug\_result'), 10, 3 );

}

function check\_for\_update( $transient ) {

if(empty($transient->checked)) return $transient;

$request\_args = array(

'id' => $this->plugin\_id,

'slug' => $this->plugin\_slug,

'version' => $transient->checked[$this->plugin\_path]

);

if ($this->license\_key) $request\_args['license'] = $this->license\_key;

$request\_string = $this->prepare\_request( 'update\_check', $request\_args );

$raw\_response = wp\_remote\_post( $this->api\_url, $request\_string );

$response = null;

if( !is\_wp\_error($raw\_response) && ($raw\_response['response']['code'] == 200) )

$response = unserialize($raw\_response['body']);

if( is\_object($response) && !empty($response) ) {

// Feed the update data into WP updater

$transient->response[$this->plugin\_path] = $response;

return $transient;

}

// Check to make sure there is not a similarly named plugin in the wordpress.org repository

if ( isset( $transient->response[$this->plugin\_path] ) ) {

if ( strpos( $transient->response[$this->plugin\_path]->package, 'wordpress.org' ) !== false ) {

unset($transient->response[$this->plugin\_path]);

}

}

return $transient;

}

function plugin\_api\_call( $def, $action, $args ) {

if( !isset($args->slug) || $args->slug != $this->plugin\_slug ) return $def;

$plugin\_info = get\_site\_transient('update\_plugins');

$request\_args = array(

'id' => $this->plugin\_id,

'slug' => $this->plugin\_slug,

'version' => (isset($plugin\_info->checked)) ? $plugin\_info->checked[$this->plugin\_path] : 0 // Current version

);

if ($this->license\_key) $request\_args['license'] = $this->license\_key;

$request\_string = $this->prepare\_request( $action, $request\_args );

$raw\_response = wp\_remote\_post( $this->api\_url, $request\_string );

if( is\_wp\_error($raw\_response) ){

$res = new WP\_Error('plugins\_api\_failed', \_\_('An Unexpected HTTP Error occurred during the API request.</p> <p><a href="?" onclick="document.location.reload(); return false;">Try again</a>'), $raw\_response->get\_error\_message());

} else {

$res = unserialize($raw\_response['body']);

if ($res === false)

$res = new WP\_Error('plugins\_api\_failed', \_\_('An unknown error occurred'), $raw\_response['body']);

}

return $res;

}

function prepare\_request( $action, $args ) {

global $wp\_version;

return array(

'body' => array(

'action' => $action,

'request' => serialize($args),

'api-key' => md5(home\_url())

),

'user-agent' => 'WordPress/'. $wp\_version .'; '. home\_url()

);

}

function debug\_result( $res, $action, $args ) {

echo '<pre>'.print\_r($res,true).'</pre>';

return $res;

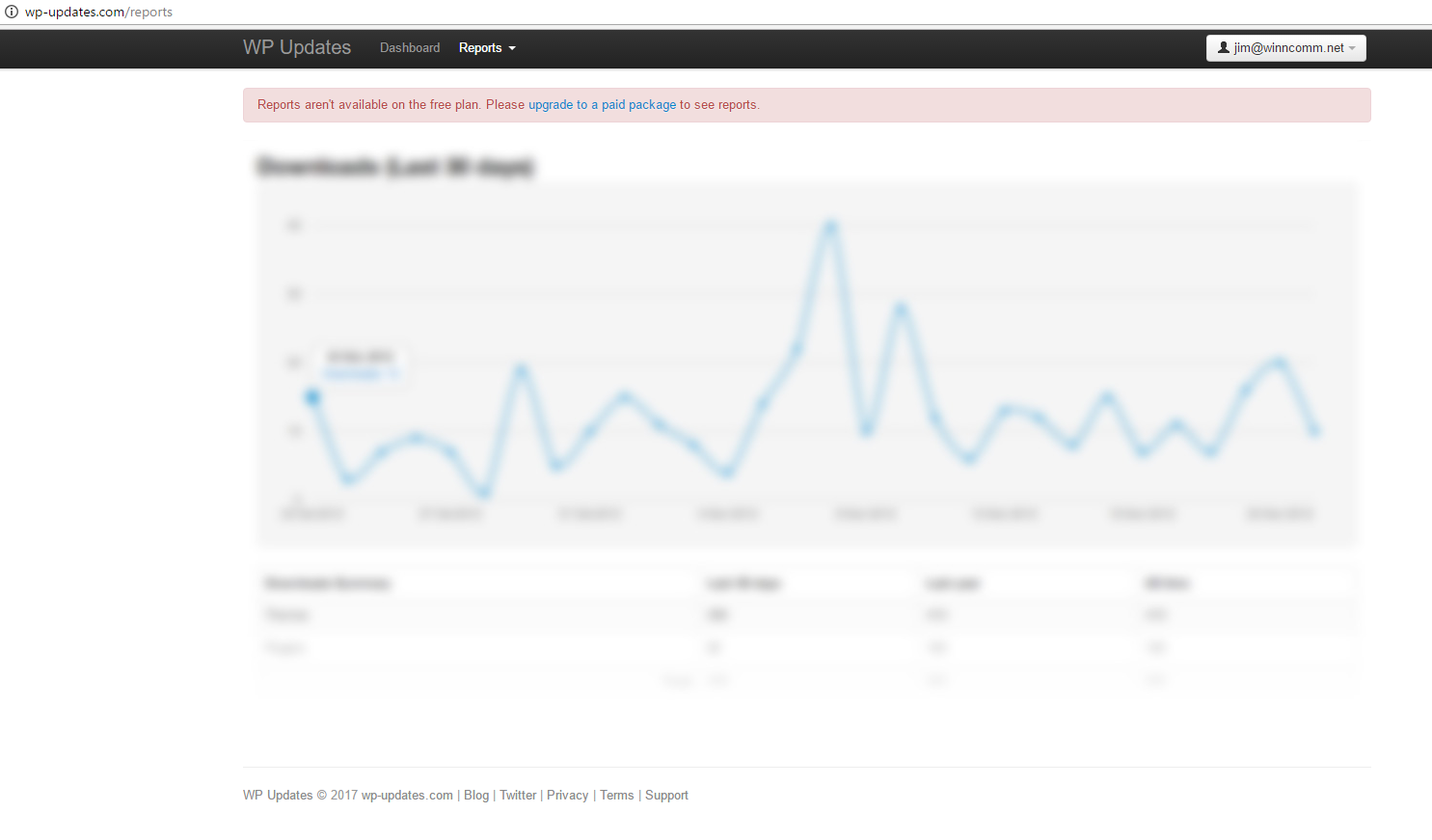
}

}

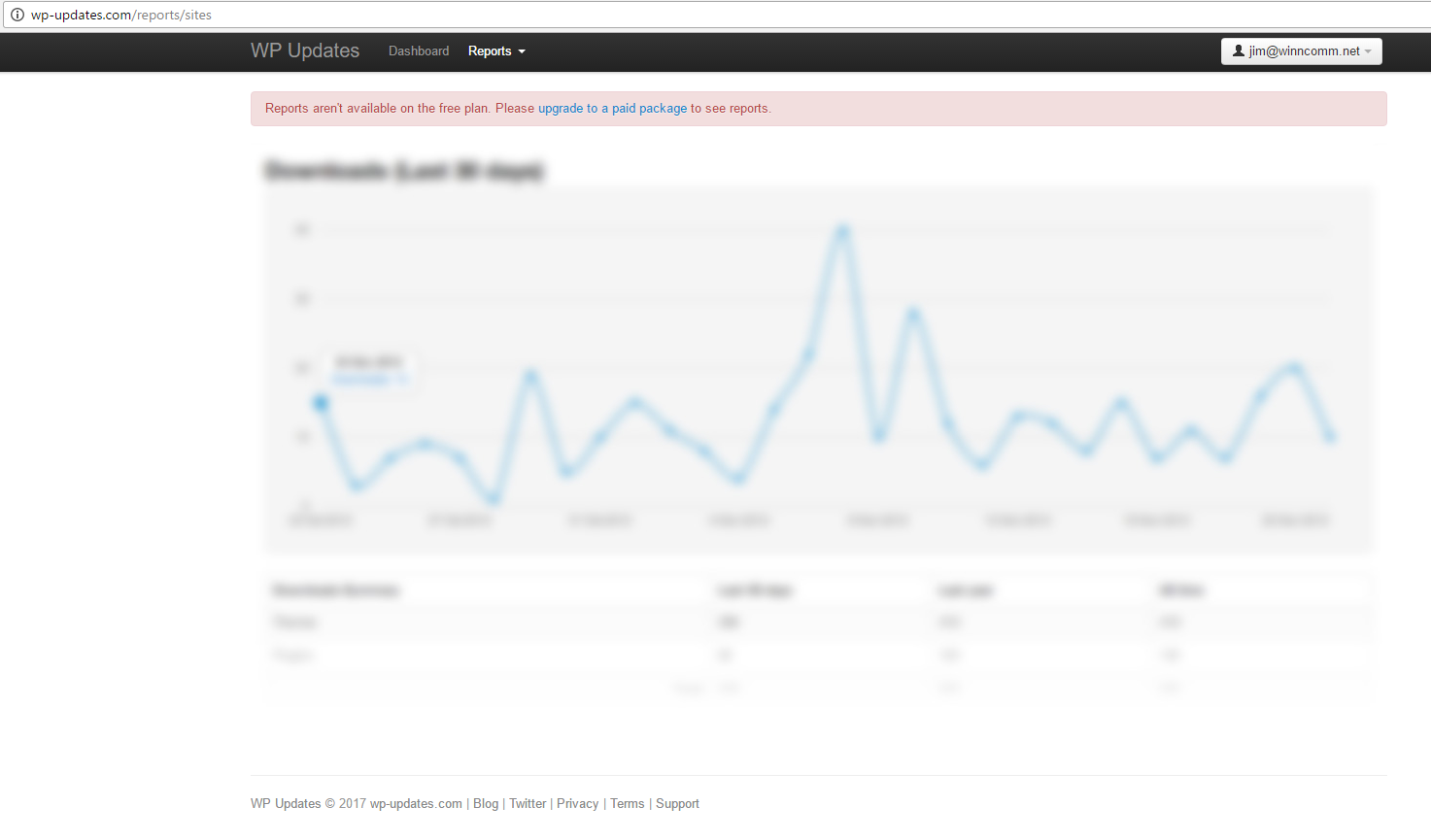
}

Reports

Reports only show the following, which I can’t see unless we purchased there and used their services, but it looks pretty simple and straightforward.

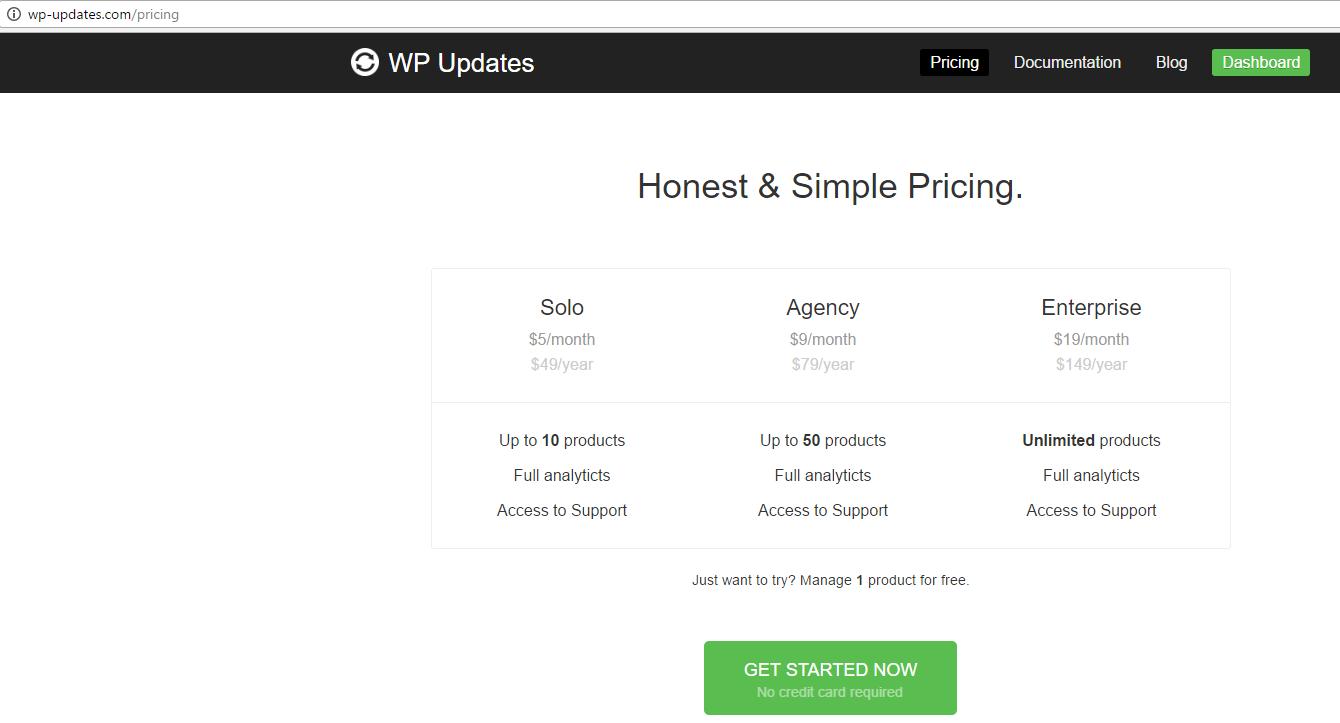


Looks like they show a sites report, of which sites are updating the plugin and from where:



Pricing

Their pricing is setup as follows:



I’m thinking we can match their pricing but we can put in a “licensing feature later.” The good thing is I can match it and out market them using WinnComm, LLC as the marketing company for this service.

Thoughts and ideas, email to me [jim@winncomm.net](mailto:jim@winncomm.net)