Интеграция PRIZM

Платёжная система PRIZM это самый простой способ приёма и отправки криптоплатежей.

Вы можете легко интегрировать PRIZM в свой интернет-магазин, обменник, биржу и многое, многое другое.

Онлайн-гайдлайн по ссылке

https://pzm.space/pzm-integration/

Интеграция платежной системы PRIZM

Для начала работы с PRIZM потребуется запуск **сетевого узла** (Нода) и **API_Servlet**

Сетевой узел

ПО может работать как на одном сервере, так и на разных, но для удобства лучше запустить на одном.

Сначала запускаем ноду и ждём пока она синхронизируется, далее приступаем к настройке модуля **PrizmAPIServlet.**

PrizmCore wallet
https://github.com/prizmspace/PrizmCore#prizmcore-wallet-download-v1103-windows-osx-linux

Easy API Gateway
https://github.com/prizmspace/PrizmCore#easy-api-gateway-prizmapiservlet

Настройка PrizmAPIServlet

Внутри архива есть файл
PrizmAPIServlet.properties

в строке
passphrase: NONE

(вместо NONE прописываем приватный ключ кошелька который будет использоваться Вашим магазином)

После заполнения полей
запускаем сервлет

run-servlet.sh

в строке
sendkey: NONE

вместо NONE прописываем свой пароль (функция отправки монет будет использовать его как дополнительную защиту от несанкционированных транзакций)

Пример реализации на PHP

Описание работы с приемом и отправкой монет, с примерами готовых функций и описанием принципов работы. Для хранения списка транзакций используется база Mysql, ниже будет приведен дамп таблицы хранения, и примеры кода для работы с ней (используя QueryBilder - это не будет проблемой)

Базовый принцип работы:

В крон-задаче висит скрипт который каждые 2-5 минут делает запрос к сервлету для получения новых транзакций по кошельку магазина. Получив список транзакций сохраняем их в свою локальную базу.

Если в базе нет операций мы делаем запуск команды без параметра, а если мы хотим получить новые транзакции, то отправляем в качестве параметра номер последней транзакции которая у нас есть.

Пример функции

```
<?php
function historyPZM($last_id = 0)
{
 if ($last_id) {
  $url = 'http://localhost:8888/history?fromid=' . $last_id;
 } else {
  $url = 'http://localhost:8888/history';
 }
 $page = '';
 $result = get_web_page($url);
 if (($result['errno'] != 0) || ($result['http_code'] != 200)) {
  $error = $result['errmsg'];
 } else {
  $page = $result['content'];
 }
 $array_new = array();
 $xcmorewrite = explode("\n", str_replace("\r", '', $page));
 foreach ($xcmorewrite as $value) {
  if ($value) {
   $array_new[] = explode(";", $value);
  }
 }
 return $array_new;
}
?>
```

Функция для получение содержимого страниц:

```
<?php

function get_web_page($url)
{
 $uagent = "Opera/9.80 (Windows NT 6.1; WOW64) Presto/2.12.388 Version/12.14";
 $ch = curl_init($url);
 curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);   // возвращает веб-страницу
 curl_setopt($ch, CURLOPT_HEADER, 0);          // не возвращает заголовки
 curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);   // переходит по редиректам
 curl_setopt($ch, CURLOPT_ENCODING, "");       // обрабатывает все кодировки
 curl_setopt($ch, CURLOPT_USERAGENT, $uagent);  // useragent
 curl_setopt($ch, CURLOPT_CONNECTTIMEOUT, 20); // таймаут соединения
 curl_setopt($ch, CURLOPT_TIMEOUT, 20);       // таймаут ответа
 curl_setopt($ch, CURLOPT_MAXREDIRS, 2);      // останавливаться после 10-ого редиректа

 $content = curl_exec($ch);
 $err = curl_errno($ch);
 $errmsg = curl_error($ch);
 $header = curl_getinfo($ch);
 curl_close($ch);

 $header['errno'] = $err;
 $header['errmsg'] = $errmsg;
 $header['content'] = $content;
 return $header;
}

?>
```

Функция для получение содержимого страниц:

Для примера можно попробовать через консоль:
curl http://localhost:8888/history

Пример скрипта обработчика крон задания для получения новых транзакций и структуры таблицы

```
CREATE TABLE `pzm_history` (
  `id` bigint(20) NOT NULL,
  `tarif_id` int(1) NOT NULL,
  `tr_id` varchar(255) NOT NULL,
  `tr_date` varchar(255) NOT NULL,
  `tr_timestamp` int(11) NOT NULL,
  `pzm` varchar(50) NOT NULL,
  `summa` decimal(16,2) NOT NULL,
  `mess` varchar(255) NOT NULL,
  `status` int(1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

***В таблицу надлежит добавить нужные ключи и автоинкремент для id**

Обработчик:

В данном примере мы получаем список новых транзакций и сохраняем в локальную базу.

Таким образом мы ведем историю всех транзакций по кошельку, и в дальнейшем будем искать их в нашей локальной базе по ключевым данным.

```
<?php
$nomer = getLastPrmHistory();
$historys = historyPZM($nomer);

foreach ($historys as $item) {
   if ($item['0'] != "No transactions!") {

// в данной строке строке идёт добавление в таблицу `pzm_history` данных с использованием INSERT IGNORE

PzmHistory::find()->insertIgnore([
          'tr_id' => $item['0'],
          'tr_date' => $item['1'],
          'tr_timestamp' => $item['2'],
          'pzm' => $item['3'],
          'summa' => $item['4'],
          'mess' => $item['5'],
          'status' => 0
      ]);
   }
}
```

```
function getLastPrmHistory()
{
// в данной строке идет поиск последней строки в таблице чтобы получить последний идентификатор
транзакции которые есть в нашей таблице

if (!empty($pzmHistory = PzmHistory::find()->orderBy('id', "DESC")->first())) {
  return $pzmHistory->tr_id;
 };
 return 0;
}

?>
```

Ваш проект должен работать с одним кошельком Prizm, поэтому всем клиентам будут выдаваться одни и те же реквизиты для пополнения внутреннего счета и хеш-идентификатор операции. Обязательно сообщаем клиенту что он должен совершить транзакцию строго по реквизитам с указанием хэш-идентификатора в комментарий платежа.

Таким образом должен быть ещё один процесс который будет анализировать новые входящие транзакции, и если в комментарии платежа есть хеш-идентификатор клиента зачислять монеты на внутренний счёт, так же для клиента нужно сделать отдельную кнопку "**Я ОПЛАТИЛ**" при нажатии которой будет производится поиск и учет новых транзакций для этого пользователя.

Вспомогательные функции, и функции отправки монет.

Получение публичного ключа для кошелька (работает только для активированных кошельков с балансом)

```
<?php

 function destinationPZM($pzm)
   {
      $url = 'http://localhost:8888/publickey?destination=' . $pzm;
      $page = '';
      $result = get_web_page($url);
      if (($result['errno'] != 0) || ($result['http_code'] != 200)) {
         $error = $result['errmsg'];
         return '';
      } else {
         $page = $result['content'];
         $haystack = "Public key absent";
         $haystack2 = "Send error!";
         $pos = strripos($page, $haystack);
         $pos2 = strripos($page, $haystack2);
         if ($pos === false AND $pos2 === false) {
            $xcmorewrite = explode(' ', $page);
            $page = trim($xcmorewrite[0]);
            return $page;
         } else {
            return '';
         }
      }
      return $page;
   }

?>
```

Получение текущего баланса кошелька:

```
<?php

 function getBalancePZM($pzm)
   {
      $ip = '*******';  // пример 192.168.1.1:9976  с указанием порта
  $url   =   'http://'.$ip.'/prizm?requestType=getAccount&account='   .
$pzm;
      $page = '';
      $result = get_web_page($url);
  //print_r($result); die;
      if (($result['errno'] != 0) || ($result['http_code'] != 200)) {
         $error = $result['errmsg'];
         return '';
      } else {
         $page = $result['content'];
         $page = json_decode($page, true);
         if ( isset($page['balanceNQT']) ) {
           return $page['balanceNQT'] / 100;
         } else {
           return 0;
         }
      }
   }

?>
```

Метод отправки монет:

```
<?php

public function payPZM($summa, $pzm, $public_key, $text)
{
 $p2 = SENDKEY;   //  это пароль вы который указывали при настройке сервлета
 $return = false;
 $url  =  'http://localhost:8888/send?sendkey='  .  $p2  .  '&amount='  .  $summa  .
'&comment='  .  urlencode($text)  .  '&destination='  .  $pzm  .  '&publickey='  .
$public_key;
 $page = '';
 $result = get_web_page($url);

 if (($result['errno'] != 0) || ($result['http_code'] != 200)) {
  $error = $result['errmsg'];
 } else {
  $page = $result['content'];
 }

 if (preg_match('/^\+?\d+$/', $page)) {
  $return = true;
 } else {
  $return = false;
 }
 return $return;
}

?>
```