Code Sample: Implementing an API REST server with CodeIgniter

The controller:

```
<?php
require(APPPATH.'/libraries/REST_Controller.php');

class Api extends REST_Controller
{
     function venue_get()
    {
           if(!$this->get('lat') || !$this->get('lng'))  $this-
>response('bad request. Please provide lat and lng parameters', 400);
           $this->load->model('venue_model');

           $params = array(
                 'lat'       => $this->get('lat'),
                 'lng'       => $this->get('lng'),
                 'radius'    => $this->get('radius'),
                 'category'  => $this->get('category'),
                 'tag'       => $this->get('tag'),
                 'limit'           => $this->get('limit'),
                 'page'      => $this->get('page'),
           );

        $venue = $this->venue_model->get($params);
        $this->response($venue, 200); // 200 being the HTTP response code
    }

     function user_get()
    {
           if(!$this->get('id'))  $this->response(NULL, 400);

           $this->load->model('user_model');
        $user = $this->user_model->get( $this->get('id') );

           if($user) $this->response($user, 200);
        else  $this->response(NULL, 404);
    }

    function user_post()
    {
        $result = $this->user_model->update( $this->post('id'), array(
            'name' => $this->post('name'),
            'email' => $this->post('email')
        ));

        if($result === FALSE)  $this->response(array('status' =>
'failed'));
        else $this->response(array('status' => 'success'));
    }

    function users_get()
    {
        $this->load->model('user_model');
           $users = $this->user_model->get_all();
        if($users) $this->response($users, 200);
           else $this->response(NULL, 404);
    }
}
```

The models:

```
<?php

class Venue_model extends CI_Model {

    function __construct()
    {
        // Call the Model constructor
        parent::__construct();
    }

      /**
     * Retrieves the venue records based on a circle area
      * The "params" argument contains all data needed, including query 
limit,
      * offset, lat/lng, radius, category, tags.
     *
     * @param  array  $params
     * @return array
     */
     function get($params=null)
     {
           if (!$params['radius']) $params['radius'] = 100;
           if (!$params['limit']) $params['limit'] = 50;
           if (!$params['page']) $params['page'] = 0;
           $params['offset'] = $params['page']*$params['limit'];

           $this->db->select('venue.id, venue.name, venue.address, 
venue.city, venue.state, venue.zipcode');
           $this->db->select('X(location) AS lat, Y(location) AS lng');
           $this->db->select('GROUP_CONCAT(DISTINCT(category.name)) AS 
category');
           $this->db->select('GROUP_CONCAT(DISTINCT(tag.name)) AS tag');
           $this->db->select('COUNT(DISTINCT(venue_review.id)) as 
number_of_reviews');

           // calculates the distance in miles
           $this->db->select("
                 (((acos(sin((".$params['lat']."*pi()/180)) * 
sin((X(location)*pi()/180))+cos((".$params['lat']."*pi()/180)) * 
cos((X(location)*pi()/180)) * cos(((".$params['lng']."-
Y(location))*pi()/180))))*180/pi())*60*1.1515)
                 as distance
           ");

           $this->db->where('venue.status', 'active');
           if ($params['category']) $this->db->like('category.name', 
$params['category']);
           if ($params['tag']) $this->db->like('tag.name', $params['tag']);


           $this->db->where("(((acos(sin((".$params['lat']."*pi()/180)) * 
sin((X(location)*pi()/180))+cos((".$params['lat']."*pi()/180)) *
cos((X(location)*pi()/180)) * cos(((".$params['lng']."-
Y(location))*pi()/180))))*180/pi())*60*1.1515) < {$params['radius']}");

           $this->db->join('venue_category', 'venue_category.venue_id =
venue.id', 'left');
           $this->db->join('category', 'category.id =
venue_category.category_id');
           $this->db->join('venue_tag', 'venue_tag.venue_id = venue.id',
'left');
           $this->db->join('tag', 'tag.id = venue_tag.tag_id');
           $this->db->join('venue_review', 'venue_review.venue_id =
venue.id', 'left');

           $this->db->group_by('venue.id');
           $this->db->limit($params['limit'], $params['offset']);
           $query = $this->db->get('venue');

           return $query->result_array();
     }
}
```

```
<?php
class User_model extends CI_Model {
    function __construct()
    {
        // Call the Model constructor
        parent::__construct();
    }

    function get_all()
    {
        $query = $this->db->get('user', 10);
        return $query->result_array();
    }

     function get($id)
     {
           $this->db->where('id', $id);
           $query = $this->db->get('user');

           return $row = $query->row_array();
     }
}
```

Code Sample: Implementing a Google Calendar library with ZendGData and CodeIgniter

The library:

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

$include_path = FCPATH.APPPATH.'libraries/ZendGdata/library';
set_include_path($include_path);

require_once 'Zend/Loader.php';
Zend_Loader::loadClass('Zend_Gdata');

Zend_Loader::loadClass('Zend_Gdata_Calendar');
Zend_Loader::loadClass('Zend_Oauth');
Zend_Loader::loadClass('Zend_Oauth_Consumer');
Zend_Loader::loadClass('Zend_Crypt');

class Gcal {

     // The library uses an Oauth authentication method
    function OAuth () {

           $CI =& get_instance();

           $CI->config->load('gcalendar');
           $CI->config->load('goauth');

           $this->credentials = $CI->config->item('credentials');
           $this->token = $CI->config->item('token');
           $this->token_secret = $CI->config->item('token_secret');

           $accessToken = new Zend_Oauth_Token_Access();
           $accessToken->setToken($this->token);
           $accessToken->setTokenSecret($this->token_secret);

           $oauthOptions = array(
             'requestScheme' => Zend_Oauth::REQUEST_SCHEME_HEADER,
             'version' => '1.0',
             'consumerKey' => $this->credentials['consumer_key'],
             'consumerSecret' => $this->credentials['consumer_secret'],
             'signatureMethod' => 'HMAC-SHA1',
             'callbackUrl' => site_url().'/gcalendar/access_token',
             'requestTokenUrl' =>
'https://www.google.com/accounts/OAuthGetRequestToken',
             'userAuthorizationUrl' =>
'https://www.google.com/accounts/OAuthAuthorizeToken',
             'accessTokenUrl' =>
'https://www.google.com/accounts/OAuthGetAccessToken'
           );

           $httpClient = $accessToken->getHttpClient($oauthOptions);
           $client = new Zend_Gdata_Calendar($httpClient, $this-
>credentials['consumer_key']);

           return $client;
     }
     function getToken($CONSUMER_KEY, $CONSUMER_SECRET) {

           $oauthOptions = array(
             'requestScheme' => Zend_Oauth::REQUEST_SCHEME_HEADER,
             'version' => '1.0',
             'consumerKey' => $CONSUMER_KEY,
             'consumerSecret' => $CONSUMER_SECRET,
             'signatureMethod' => 'HMAC-SHA1',
             'callbackUrl' => site_url().'/gcalendar/access_token',
             'requestTokenUrl' =>
'https://www.google.com/accounts/OAuthGetRequestToken',
             'userAuthorizationUrl' =>
'https://www.google.com/accounts/OAuthAuthorizeToken',
             'accessTokenUrl' =>
'https://www.google.com/accounts/OAuthGetAccessToken'
           );

           $consumer = new Zend_Oauth_Consumer($oauthOptions);

           return $consumer;
     }

     function createEvent($gdataCal, $eventArray){

           $newEvent = $gdataCal->newEventEntry();

           $newEvent->title = $gdataCal->newTitle($eventArray['title']);
           $newEvent->where = array($gdataCal-
>newWhere($eventArray['where']));
           $newEvent->content = $gdataCal->newContent($eventArray['desc']);

           $when = $gdataCal->newWhen();
           $when->startTime =
"{$eventArray['startDate']}T{$eventArray['startTime']}:00.000{$eventArray['
tzOffset']}:00";
           $when->endTime =
"{$eventArray['endDate']}T{$eventArray['endTime']}:00.000{$eventArray['tzOf
fset']}:00";
           $newEvent->when = array($when);

           //upload the even to the calendar server
           //a copy of the event as it is recorded on the server is
returned
           $createdEvent = $gdataCal->insertEvent($newEvent);

           return $createdEvent->id->text;
     }

     function getEvent($gdataCal, $eventId)
     {
       $eventId = str_replace("/", "",  strrchr($eventId, "/"));

       $query = $gdataCal->newEventQuery();
       $query->setUser('default');
       $query->setVisibility('private');
       $query->setProjection('full');
       $query->setEvent($eventId);
```

```
        try {
            $eventEntry = $gdataCal->getCalendarEventEntry($query);
            return $eventEntry;
        } catch (Zend_Gdata_App_Exception $e) {
            echo "Error: " . $e->getMessage();
            return null;
        }
    }

    function updateEvent($gdataCal, $event_id, $data) {

            $event_id = str_replace("/", "",  strrchr($event_id, "/"));
            $event = $this->getEvent($gdataCal, $event_id);

            if (!$event) return false;

            $event->title = $gdataCal->newTitle($data['title']);
            $event->where = array($gdataCal->newWhere($data['where']));
            $event->content = $gdataCal->newContent($data['desc']);

            $when = $gdataCal->newWhen();
            $when->startTime =
"{$data['startDate']}T{$data['startTime']}:00.000{$data['tzOffset']}:00";
            $when->endTime =
"{$data['endDate']}T{$data['endTime']}:00.000{$data['tzOffset']}:00";
            $event->when = array($when);

            $event->save();

            return true;
    }

    function deleteEvent($client, $event_id) {

            $event_id = str_replace("/", "",  strrchr($event_id, "/"));
            $event = $this->getEvent($client, $event_id);
            if (!$event) return false;
            $event->delete();
            return $true;
    }
}

?>
```

The controller:

```
<?php

Class Gcalendar extends CI_Controller {

    function __construct(){
            parent::__construct();
            $this->load->library('Gcal');
            $this->config->load('gcalendar');

            $this->credentials = $this->config->item('credentials');
            if ($this->credentials['api_username'] != $_POST['api_username']
|| $this->credentials['api_password'] != $_POST['api_password'])
exit('Invalid Credentials');
     }

     function get_token()
     {
            $consumer = $this->gcal->getToken($this-
>credentials['consumer_key'], $this->credentials['consumer_secret']);
            // Multi-scoped token.
            $SCOPES = array( 'http://www.google.com/calendar/feeds');
            if (!isset($_SESSION['ACCESS_TOKEN'])) {
                  $_SESSION['REQUEST_TOKEN'] = serialize($consumer-
>getRequestToken(array('scope' => implode(' ', $SCOPES))));
            }
            $consumer->redirect(array('hd' => 'default'));
     }
     function access_token()
     {
            $consumer = $this->gcal->OAuth($this-
>credentials['consumer_key'], $this->credentials['consumer_secret']);
            if (!isset($_SESSION['ACCESS_TOKEN'])) {
              if (!empty($_GET) && isset($_SESSION['REQUEST_TOKEN'])) {
                  $_SESSION['ACCESS_TOKEN'] = serialize($consumer-
>getAccessToken($_GET, unserialize($_SESSION['REQUEST_TOKEN'])));
              }
            }

            $token = unserialize($_SESSION['ACCESS_TOKEN']);
            $fp = fopen(APPPATH.'config/goauth.php', 'a');

            // Write the token into a configuration file
            fwrite($fp, "\n");
            fwrite($fp, '$config["token"] = '.$token->getToken(). '; //
updated at '.date('m/d/Y H:i:s'));
            fwrite($fp, "\n");
            fwrite($fp, '$config["token_secret"] = '.$token-
>getTokenSecret(). '; // updated at '.date('m/d/Y H:i:s'));
            fclose($fp);
     }

     function addEvents(){

            $client = $this->gcal->oauth();

            $events = $_POST['events'];
            foreach ($events as $event) {
                  try {
                        $google_id = $this->gcal->createEvent($client,
$event);
                  } catch (Zend_Gdata_App_Exception $e) {
                        echo "Error: " . $e->getMessage();
                  }
            }

            echo $google_id;
     }
     function updateEvent(){

          $client = $this->gcal->oauth();

          $events = $_POST['events'];
          $event_id = $_POST['event_id'];
          foreach ($events as $event) {
               try {
                    $this->gcal->updateEvent($client, $event_id,
$event);
               } catch (Zend_Gdata_App_Exception $e) {
                    echo "Error: " . $e->getMessage();
               }
          }
     }

     function deleteEvent() {

          $client = $this->gcal->oauth();
          $event = $_POST['event'];
          foreach ($event as $event_id) {
               $event = $this->gcal->deleteEvent($client, $event_id);
          }
     }

     function getEvent() {

          $event_id = $_POST['event_id'];
          $client = $this->gcal->oauth();
          $event = $this->gcal->getEvent($client, $event_id);

          if (!$event) echo "0";
          else echo "1";
     }

     public function listEvents() {

          $client = $this->gcal->oauth();

          $eventFeed = $client->getCalendarEventFeed();

          foreach ($eventFeed as $calendar) {
               echo "<li>" . $calendar->title .
                    " (Event Feed: " . $calendar->id . ")</li>";
          }

     }
}
?>
```