**Technical Document**

*For*

**WOM**

*Word of mouth Referrals*

Version 1.0

Sept 2th, 2013

*Submitted to*

Mr. Tom Knauss



*Prepared by*

Analysis Team Of

Avion Technology, Inc.

Visit us at: [www.avionechnology.net](http://www.avionechnology.net)

Contents

[1. Introduction: Codeigniter Architecture 3](#_Toc365627000)

[1.1 Chosen Technologies 3](#_Toc365627001)

[1.2 General Overview (Codeigniter Based) 3](#_Toc365627002)

[1.2.1 Codeigniter Default 4](#_Toc365627003)

[2 PayPal Payment Gateway 4](#_Toc365627004)

[3 Price Upload 6](#_Toc365627005)

[3.1 Codeigniter Library: Spreadsheet\_Excel\_Reader 6](#_Toc365627006)

[4. Reports 7](#_Toc365627007)

[5. Database schema and relationship diagram 7](#_Toc365627008)

# Introduction: Codeigniter Architecture

* MVC based – see more details below

## 1.1 Chosen Technologies

* Programming language: PHP
* Framework: Codeigniter
* Database: Mysql
* Script Language : Javascript, Jquery

## 1.2 General Overview (Codeigniter Based)

* Diagram below represents basic architecture driven by Codeigniter.



* The index.php serves as the front controller, initializing the base resources needed to run Codeigniter.
* The Router examines the HTTP request to determine what should be done with it.
* If a cache file exists, it is sent directly to the browser, bypassing the normal system execution.
* Security. Before the application controller is loaded, the HTTP request and any user submitted data is filtered for security.
* The Controller loads the model, core libraries, helpers, and any other resources needed to process the specific request.
* The finalized View is rendered then sent to the web browser to be seen. If caching is enabled, the view is cached first so that on subsequent requests it can be served.

### Codeigniter Default

* By default, CI (version 1.7.1) presents you with the following structure:
* system
* application
  + - cache
    - Codeigniter
    - database
    - fonts
    - helpers
    - language
    - libraries
    - logs
    - plug-in
    - scaffolding
* We will be creating an assets folder on root of the project which contains folders like css, script, images etc.

## PayPal Payment Gateway

* We will use a PayPal payment gateway to integrate payment in site.
* We will use Codeigniter library to integrate the payment.
* Below are the required points while coding for the same.
* The amount for which user needs to pay should be passed to paypal.
* Merchant account details along with other information have to be passed to the paypal API.
* The payment process should handle any unwanted condition occurs like power failure/connection failure while making a payment.
* IPN integration is necessary for the Instant Payment Notification.
* After testing on sandbox the PayPal will be live mode.
* Below are the Codeigniter class library for integration :
* paypal\_class : we will use all the functions defined in this class for posting to paypal.
* The PayPal integration is divided in three parts :
* **View**: This will contain all the product related information. This will be having all the dynamic data which user can manage.
* **Controller**: Controller will include the library to use for paypal. View will post all the dynamic data in controller and controller will send the data to paypal using library and model.
* **Model**: Model will insert data in database if needed.
* In checkout controller (or whatever controller name is) where you implement your checkout system, add method PayPal as below:

|  |
| --- |
| public function paypal() {  $this->load->library('paypal\_class');  $this->paypal\_class->paypal\_url = 'https://www.sandbox.paypal.com/cgi-bin/webscr'; // testing paypal url  //$this->paypal\_class->paypal\_url = 'https://www.paypal.com/cgi-bin/webscr'; // paypal url  $this->paypal\_class->add\_field('currency\_code', 'CHF');  $this->paypal\_class->add\_field('business', $this->config->item('bussinessPayPalAccountTest'));  //$this->paypal\_class->add\_field('business', $this->config->item('bussinessPayPalAccount'));  $this->paypal\_class->add\_field('return', $this->base.'/checkout/success'); // return url  $this->paypal\_class->add\_field('cancel\_return', $this->base.'/checkout/step4'); // cancel url  $this->paypal\_class->add\_field('notify\_url', $this->base.'/validate/validatePaypal'); // notify url  $totalPrice = $this->session->userdata('totalPrice');  $this->paypal\_class->add\_field('item\_name', 'Testing');  $this->paypal\_class->add\_field('amount', $totalPrice);  $this->paypal\_class->add\_field('custom', $this->session->userdata('orderId'));  $this->paypal\_class->submit\_paypal\_post(); // submit the fields to paypal  //$p->dump\_fields(); // for debugging, output a table of all the fields  exit;  } |

* First load PayPal library and set parameters for redirection (paypal\_url)
* Then add some fields for posting to PayPal (add\_field)
* After that submitted post to PayPal (submit\_paypal\_post()) How system works? When customers click on the “GO TO PAYPAL” button, system just call paypal(); method, and customer will be forwarded to PayPal site.
* Now, we need to program one more script for validating data when PayPal return us the data.
* We will create new controller validate, and in this controller method validatePaypal();

|  |
| --- |
| public function validatePaypal() {  $this->load->library('paypal\_class');  $this->paypal\_class->paypal\_url = 'https://www.sandbox.paypal.com/cgi-bin/webscr'; // testing paypal url  //$this->paypal\_class->paypal\_url = 'https://www.paypal.com/cgi-bin/webscr'; // paypal url  if ($this->paypal\_class->validate\_ipn()) {  $orderId = trim($\_POST['custom']);  $itemName = trim($\_POST['item\_name']);  // put your code here  }  break;  } |

* So, if PayPal pass throw validate\_ipn() method you can add that payment to database.
* Below is the system workflow (in few steps):
* User books a tour
* Goes to checkout (system count price….)
* User clicks at “GO TO PAYPAL” button
* System redirects customer to PayPal
* Pays bill at PayPal
* PayPal contact script (validate/validatePaypal) and send the data
* PayPal redirects customer to return url

## Price Upload

* Below are the Codeigniter libraries we will use to implement the price upload in CSV files

### Codeigniter Library: Spreadsheet\_Excel\_Reader

* This library is used to read the data in a CSV file.
* There will be a limit on the size of CSV to be uploaded.
* Below are the steps for the excel sheet read and upload:

#### Upload CSV

#### 

* We will first upload a proper format CSV file form the admin panel.
* We will use jquery as well as PHP validation to validate the file.
* The file should be validated on below points :
* It should be a CSV file.
* The size of file should not exceed 2MB.
* The maximum amount of data rows in file is 5000.
* The validated file will be uploaded in a folder at server.
* The content of the excel sheet will be read using excel sheet reader library in Codeigniter.
* Using the library all the data will be fetched in an array.
* Then using an insert query in Mysql all data will be inserted in database tables.
* Below are the Codeigniter library parameters used to upload file:

|  |
| --- |
| $config['upload\_path'] = './uploads/'; $config['allowed\_types'] = 'csv|CSV'; $config['max\_size'] = '100'; $config['file\_name'] = date(“Y-m-d H:i:s”); $config['encrypt\_name'] = TRUE;  $config['remove\_spaces'] = TRUE; |

* Security Majors :
* Rename the files uploaded to the current datetime.
* After the data is uploaded on database delete the file from the server.

## 4. Reports

* Admin can download all the data in an excel sheet format.
* The reports section will have a common search dropdown of dates so that admin can download data from particular date range.
* Because the data is very large, admin cannot download whole database at a time.
* The search option will have a date range selector as: From and to.
* Below are the libraries used for CI excel sheet reader/write.
* Codeigniter library/helpers used for the reports download functionality:
* Spreadsheet\_excel\_reader
* Upload
* Download
* The upload library will be used to upload and validate the CSV file.
* Download library is used to download the file.
* Spreadsheet excel reader is used to select data and import that in an excel or CSV file.

## 5. Database schema and relationship diagram

* The database will be developed in Mysql.
* Below are the important tables of the system :

1. Tbl\_hotels
2. Tbl\_room\_types
3. Tbl\_travelers
4. Tbl\_booking
5. Tbl\_admin
6. Tbl\_destinations
7. Tbl\_pricing
8. Tbl\_payments
9. Tbl\_weeks\_of\_travel
10. Tbl\_special\_offers
11. Tbl\_add\_on
12. Tbl\_contactus
13. Tbl\_hotel\_videos
14. Tbl\_hotel\_pictures
15. Tbl\_faq

* The diagram below explains the relationship between the tables and fields.
* The master tables will have their primary keys as the foreign keys in other tables.
* For example: tbl\_hotel will have its primary key as the foreign key in tbl\_hotel\_pictures and tbl\_hotel\_videos etc.
* There are some independent table which do not have any relation to other tables like : tbl\_contact\_us,tbl\_faq .

