ERP For GIS Technical Documentation

Introduction

This is the primary documentation for the ERP for GIS (IIT Roorkee).

The application is written using PHP, and is written for mySQL, although other databases could be bundled in easily. Other technologies used include Google Maps API for mapping, RedBeansPHP as an ORM layer, LightOpenID (php) to allow open-id authentication from the Brihaspati Server.

Folder Structure

The application is written using MVC (Model-View-Controller) pattern. The various folder of the application are described below :

- **cache/** - Holds cached responses for various queries. Not fully used as of now.
- **controllers/** - Holds all the controllers
- **css/** - Stylesheets
- **includes/** - Most of the core framework
- **js/** - Javascript
- **models/** - Database Models, mostly validation
- **modules/** - PHP Modules for various tasks, such as ORM, Session, and Helpers
- **views/** - Views for MVC pattern.
- **index.php** - The entry point for the application.

Basic Flow

The basic flow of the application involves the index.php file loading up the framework. Which in turn, loads the configurations, basic functions required by the application, and finally sends the control to “controller.php”. This file calls the required controller. The controller name is decided as per the following :

The url is assumed as **http://localhost:8080/index.php/controller/action**
Please note that, this leads to the application installation only working under a root directory of the virtual-host. You should, create a new virtual host under apache, running on a different port to test this application. Ability to run inside folders will be implemented soon.

Once the name of the controller and action have been decided, the application tries to create an instance of that controller. This calls the autoload function, which (depending on the class name) includes the required files for that controller.

Once these files have been included, the application creates an instance of the controller, and check if the controller has a method with the same name as the action. It does this by using the

Reflection API available in PHP, which allows one to introspect classes and objects. If such a function is found, it is called.

Output from an action is assumed to be a “string” which is then echoed back to the client. The output is generated using “render” functions described in the base controller class (which is inherited by all the controllers). The render function features a mechanism, which allows one to easily divide the view into portions, and call render for each one of them individually. This was mostly inspired from the way Kohana and CodeIgniter frameworks render views (as partials).

Installing

Installation should be easy enough. The following steps must be followed :

1. Copy the structure to either :
    a. /var/www or equivalent folder for apache root
    b. some other folder, which is served by apache using a virtual host
2. Edit the configuration in **includes/config.php**. This will include the database settings, and the SITE_DOMAIN config, which is used by OpenID
3. Visit the application in your browser.

Other stuff

The application is written using PHP, so familiarity with the language will help a lot. There is no db schema supplied as of the moment as RedBeans creates it on the fly (ORM). This helps simplify a lot of overhead for writing SQL queries as such. LightOpenId authentication is fully working at the moment, and requires you to first create an account on the Brihaspati site, before logging in to the application.

The application redirects you to the open-id server, and validates the token received after authentication. Most of the application is documented inline as well, so take a look at that as well.