

## PHP Backend

### Directory Structure

The directory structure of the code is as follows:

- / (home directory)
  - **ajax** : Contains all scripts whole sole purpose is to respond to AJAX requests.
  - **auth**: Contains files related to login/signup of the website. Scripts for login through Facebook and Twitter are also here.
    - **twitter\_oauth** : Contains files related to Twitter login.
  - **css** : Contains all CSS scripts
    - **fonts** : Fonts used by the site.
  - **db** : Contains database backups
  - **fb\_php\_sdk** : Contains files of Facebook PHP SDK ( Visit <https://github.com/facebook/php-sdk> )
  - **images** : Images used by the CSS. User uploaded images are NOT stored here.
  - **include** : Commonly included utility scripts are contained here.
  - **js** : Contains Javascript files
  - **mail** : Contains scripts for sending emails.
    - **images** : Contains images used by mails
    - **phpmailer** : Contains library files of PHPMailer (<http://phpmailer.worxware.com/> )
  - **uploads** : Contains user uploaded pics
    - **food\_pics** : Contains pictures of food items
    - **profile\_pics** : Contains uploaded profile pictures of users.

### Configuration Settings

All configuration settings are stored in 'include/config.inc.php' in the form of DEFINE constants. These constants will be available to any script which includes the above file at the beginning.

Setting	Description
IS_LOCAL_SERVER	Set to true when testing on local server, false when code is running on the main server. This helps to quickly switch between local server and the main server.
BASE_URL	URL of the home directory (the one containing index.php).
MYSQL_USERNAME	MySQL username.
MYSQL_PASSWORD	MySQL password.
MYSQL_DB	MySQL database name.
ABS_PATH_TO_HOME	Absolute path of the home directory relative to root directory of the operating system of the server.
MAX_UPLOAD_SIZE	Max. permissible size of image uploaded in upload_images.php
FB_APP_ID	Application ID of foodiescompass Facebook app.

FB_APP_SECRET	Application secret of foodiescompass Facebook app.
TWITTER_APP_ID	Application ID of foodiescompass Twitter app.
TWITTER_APP_SECRET	Application secret of foodiescompass Twitter app.
TWITTER_OAUTH_CALLBACK	Required setting in Twitter App Dashboard. Must exactly match the one entered in Twitter App Dashboard.
IMG_DIR	Directory where food images are stored.
MIN_AGE	Minimum age to signup on Foodies Compass.
SALT1	Used for some encryption.


**Session Management:** Include 'include/session.inc.php' at the beginning of any script for session management.

Action	Function
Starting a new session	<i>start_session(user_id)</i>
Ending an existing session	<i>end_session()</i>
Checking if a browser session exists	<i>is_logged_in()</i>

**General Utility Functions:** Contained in 'include/lib.inc.php'.

Function	Description	Arguments	Returns
getUniqueld([\$seed])	Returns a 6-character unique unique_id.	\$seed [optional] Any string value that can be used as a seed. Helps to obtain a random value quickly.	A six character of the form 'zA_f3a' that no other user has unique_id.
curl(\$url)	Initiates a curl request when a string result is expected.	\$url : A url including the GET parameters Ex. 'http://foodiescompass.com/give_me.php?id=47'	Result string of the request.
page_path(\$identifier)	Define identifiers for pages here. Helps to change filenames easily in the future without changing URLs throughout the site	\$identifier (string) : Identifying string used for the page	Path of the page relative to the home directory;
get_page_url(\$page, [\$params])	Returns the absolute page url alongwith	\$page (string): Identifier of the page as defined in page_path	Returns absolute URL of the page alongwith GET parameters.

	parameters, if supplied.	\$params (array) : An associative array, each item representing a parameter having parameter name as item-key and parameter value as item-value.	Eg. get_page_url("food",array("eid"=>4)) returns <b>http://foodiescompass.com/beta/food.php?eid=4</b> if BASE_URL is set to "http://foodiescompass.com/beta"
is_an_integer(\$input)	Checks if the value of a variable (type may be string or numeric) is resolvable into an integer.	\$input: Value of variable to be checked	True if numeric, false otherwise.
render_follow_button(\$id)	Render a follow/unfollow button for a user	\$id : user_id of user to be followed	If user is not logged in, a 'follow' button is rendered clicking on which shows a popup which prompts the user to login to follow the user.  If the user is logged in, and if id passed is that of logged in user, no button is rendered.  If user is logged in and id passed is not that of logged in user, 'follow' or 'unfollow' button is rendered, depending on whether or not the current user is already following the other user.
render_like_pair(\$eid)	Renders a like/Unlike button for a food item	\$eid: food_items_id of the food being liked	
time_text(\$past_timestamp)	Use this to get textual description of a past time with respect to the current time	\$past_timestamp : A UNIX timestamp of a past time	Returns a textual description of a past time with respect to the current time. Eg. returns results like 'A few seconds ago', '1 hour ago', '3 hours ago', etc. [Note: It takes care of the grammatical plurality/singularity of the

			number used. ]
	Updates user_activity table of database with the user_activity	\$activity_type: Defined in flags.inc.php	-
personalize(\$user_id, \$second_person, \$third_person)	Decides whether a second person pronoun should be used or a third person pronoun. E.g. "This user does not have any badges" should be written as "You don't have any badges", if the user being talked about is the logged in user himself.	<p>\$user_id: user_id of the user who is the subject of the sentence.</p> <p>\$second_person : Second person string to be used, if \$user_id matches user_id of the logged in user.</p> <p>\$third_person: Third person string to be used if \$user_id does not match user_id of the logged in user.</p>	<p>\$second_person, if \$user_id matches user id of the logged in user , otherwise \$third_person.</p> <p>Usage:</p> <pre>&lt;span&gt; &lt;?php echo personalise(\$userdata['id'], "You do", "This user does"); ?&gt; not have any badges.&lt;/span&gt;</pre> <p>results in</p> <pre>&lt;span&gt;You do not have any badges&lt;/span&gt; if \$user_id is equal \$_SESSION['id']</pre>
render_star_panel(\$points, \$out_of)	Renders a star panel  (used by users for giving review )	<p>\$points: Score</p> <p>\$out_of: Max. possible score</p>	HTML code for the star panel, with number of stars highlighted obtained by the formula $\text{floor}(\frac{\$points}{\$out\_of} * 5)$

### Inter-script communication

Communication between scripts is done by means of standard flags defined in 'include/flags.inc.php'. The comments in the file are self sufficient to describe the purpose of each flag.

### General structure of a PHP file

The general structure used by most PHP files of the site is as follows:

```
<?php

// All includes

require_once('config.inc.php');
require_once('session.inc.php');
require_once("connection.inc.php");
require_once("flags.inc.php");
require_once("lib.inc.php");

.
.
.

// All defines
define(.....
define(.....
.
.
.

?>
<html>....

                // HTML Markup
</html>
<?php
    include('php_js.php');        // PHP generated Javascript
    .
    .

?>
```

## **Frontend**

All UI related scripts are in 'include/php\_js.php'. The PHP script generates PHP code depending on the value of \$TITLE parameter, which keeps track of the page currently the user is in.