**WEB APPLICATION AND SITE CHECKLIST**

**Project: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Date Issued: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Developer: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**General (all projects)**

* ! Use Bootstrap for user-end and CMS sites.
* ! Follow the exact design in the provided PSD file.
* Test the site on Windows, Mac, Android and iOS devices. Use Google Chrome’s Device Simulator for fast adjustments, then finally test on ACTUAL devices.
  + All pages have been tested for Desktop (IE 9+, Firefox and Chrome, side by side)
  + All pages have been tested for Tablet (iOS and Android)
  + All pages have been tested for Mobile (iOS and Android)
* No distorted images.
  + Static images (Logos, buttons, etc.) should be resized proportionally in Photoshop.
  + Dynamic images (Uploaded photos and thumbnails by Admin or End User) should be resized from SERVER SIDE using our function.resize.php.
* Parallax scrolling should be tested on a Mac.
* Use http://www.fontsquirrel.com/tools/webfont-generator OR Google Fonts on main body and other fonts, just choose simple but nice fonts. NO SYSTEM FONTS.
* No wrong grammar/ spelling, especially in notifications/ alert messages.
  + Use our standard messages sheet.
  + Use the correct type of notification/alert message (Success, Info, Warning and Danger)
  + Error messages for forms
    - It should be clear that an error has occurred
    - It should be clear what the user must do to correct the error
    - It should be clear for the user where the error was found in the form (List down the fields and change their border colors to red)
    - All errors should be displayed at the same time
    - Do not give any hint about any security event (eg. wrong: “Password is invalid!” correct: “Username or password incorrect.” or “Email and password do not match”
* Use only English Naming Conventions for all programming languages.
* Every interactive element should have a hover effect.
* Fonts, margins, paddings and pop-up styles are consistent.
* Margins of text are not too close to edges.
* Mouse-over images and background images of modal windows are PRELOADED.
* No old-style JS alerts. Use at least the jQuery UI (https://jqueryui.com)
* Transparent GIFs/ preloaders have the same matting as its background if any.
* JPEG quality for photos are in MEDIUM or lower to decrease file size as much as possible.
* Use PNG if there isn't many shades of colors in the image. (compare file size to JPEG version)
* In general, use JPGs for photos, and PNG or GIF for logos and text so they are not blurry. Logos should always be sharp.
* NO horizontal scrolling on 1100px browser/ monitor. (resize browser to test)
* Smooth transitions for carousel, slide show etc.
* All string literal transfers passed our test string: [~!@#$%^&\*()\_+[]{}:";',./<>?`1234567890éåÑé\](mailto:~!@#$%^&*()_+[]{}:";',./<>?`1234567890éåÑé\)
* Lorem.Ipsum-123\_abc@abc-DEF.com.ph should pass e-mail validation.
* Use our test string for everything, especially CMS/Excel Export, E-Mail, etc.
* Show an appropriate preloader for ALL AJAX functions. ALL AJAX calls should not cache files (add cache:false and rand:<random number> in jQuery ajax parameters)
* Line breaks are filtered when data is exported to XLS or CSV. Replace \n with \r or even just remove them.

**MySQL (all projects)**

* Primary ID should be BIGINT; Social Network IDs (FB ID, etc.) should be VARCHAR
* Our MySQL naming convention:
* Table names: projectname\_\_table\_name (all lowercase, no underscores in the project name part)
* Column names: PascalCase (First letter per word is in uppercase)
* Columns should be NON NULL.
* Create INDEX to searchable columns to improve performance.
* If the execution is still very slow, use temporary tables.
* For body of articles, use FULLTEXT search index for more sensible results.

**SEO (except Facebook Apps and CMS)**

* Google Analytics is implemented (Except for CMS)
* Each page has an HTML document title (Unique and descriptive HTML titles e.g. <Section Name> | <Site>)
* Site should have a Favicon.
* Use h1 tags on titles of the page for SEO.
* META/ Social Network META tags are applied (Title, Content Type, Description, Keywords, URL, Image, and data for Twitter Cards)
* SEO-friendly URLs are applied.
* For AJAX-driven pages, please implement Hashbang (#!) and Escaped Fragments (?\_escaped\_fragments\_). Add anchor tags to ALL links; Include the hasbang If necessary.
* All important images in the site must have ALT attributes.  This should include at least the Main Logo, articles images, slideshow, and product images.  Article images may contain the caption or article title, product images must contain the product name.
* Use Anchor tags for links, not just DIVs with click functions.
* Use IMG tags for important images, not DIVs.

**Facebook Applications (Page Tab App or Full FB App)**

* Facebook Flow: Like Page (check if page is liked via Signed Request) -> Authorization (JS library for the rest) -> Registration (fields prefilled with possible data from FB) -> App
* Use the latest Facebook PHP SDK. No JS.
* Do not use SESSIONs. (Mac Safari 5 & 6 issue). Do not pass the Facebook ID around the app. Call $facebook->getUser() function to retrieve the Facebook ID.
* Privacy Policy and Terms & Conditions are set in App Settings, and links appear in ALL pages.
* Sandbox is turned ON on development/testing stage.
* AJAX-driven.
* Show initial preloader so that the App is not blank while being loaded.
* Create handlers for unauthorized actions (ex: when user tries to post via App but has not authorized the App to post on the user's behalf).
* Irretrievable e-mails should be saved as <Facebook ID>@facebook.com. For other data, save them as blank strings. No NULL values.
* Set Admin Roles.
* No scrollbars in the FB iFrame. (Use FB.Canvas.setSize function).
* Focus on alert messages (smooth slide for better effect).
* Add “?ref=ts” to application URLs to avoid redirecting to the Mobile Version of Facebook when browsing the on your phone.
* Always use the latest Graph API version

**Security (all projects)**

* **(New)** All projects that store personal information should be encrypted (Use our crypt.php)
* All projects should have a robots.txt file to disallow the ff. directories:
  + CMS / Admin
  + Uploads folder for user-submitted content
  + Callback URL of payment gateway
* Validation in client side is optional but required in server side.
* Upon successful login, generate a random access token and store in the database. Store the access token and user name in PHP session.
* All pages that require login must run a function that checks the PHP sessions (username & access token) and compares them with the database. If this fails, logout the user. If successful, return the username as a PHP variable, readable by the current script.
* All pages that display user information, such as profile page, etc… must get the username using the above method. Must never just pass around the username as a parameter since it can easily be faked.
* Forms (Input textfields / TinyMCE)
  + Strip tags in PHP if HTML is not necessary
  + If HTML support is needed (e.g. TinyMCE), filter the following in PHP before inserting into the DB
    - <script> tags (this allows hackers to execute JS functions in the site, to collect data or just to mess up the page)
    - <iframe> tags (same as above, and allows them to track performance of the page)
    - <img> tags that have URL as SRC (optional, case-to-case basis) (allows them to track performance of the page)
    - <embed> tags (optional, case-to-case basis) (allows them to track performance of the page)
    - <audio> tags (optional, case-to-case basis) (allows them to track performance of the page, or just make it annoying)
* Keep access and passwords secure.
* Passwords should always be stored like this: MD5 (PASSWORD + SALT) on the server.
* REMOVE staging DB credentials when uploading to live server.
* Always make your PHP scripts secure.
  + Protect against SQL injection (use PDO. If PDO is not possible, use mysqli\_real\_escape\_string).
  + Store data in the database as is. Use PHP htmlentities and/or stripslashes only when showing data in front end.
  + For form to mail scripts, please always check for HTTP REFERER to make sure that the request comes from our staging server of the client's domain (always store HTTP\_REFERER, client IP and date/time on all records in the DB)
* For file/ photo user uploads in apps/ sites/ CMS
  + Always check the file type after the file is uploaded in PHP, even if you have JS validation. JS validation can easily be bypassed. File/MIME type should be checked by PHP, not just file extension.
  + Always assign a unique ID and rename the uploaded file.
  + Always put the REQUIRED .htaccess file in all directories where users (and CMS) can upload photos. This will make sure any PHP files that get uploaded will not run.
  + Images and uploaded files must have their own separate folders.