# TECHNICAL ARCHITECTURE OVERVIEW

# Project characteristics

* Technology : MEAN stack CMS
* Version: 1.0.0
* Hosting requirements:
* Browser Compatibility: IE8+, Chrome 23+, Firefox 17+, Safari 5+

# Project presentation

Eagle Manager is a MEAN stack based CMS that represents a project organization which facilitates the communication between developers and project managers. The system contains possibilities for creating, reviewing, updating and deleting projects depending on user access level. Every single project comes out with individual chat panel which includes all participants in the conversation. All conversations in which the logged in user takes place are represented on one single view and that helps him to orientate during the work process. Real time and also email notifications are added in order to mark the important messages which a separate group of users receives.

# Structure

Description of the structure disregarding modules and technique.

Eagle Manager is a standalone web application that is installed under <http://www.lahalle.com/lh-magazine>

This is the main endpoint to access the root of the web application.

The site has a top navigation. There are 4 main navigation parts with active tab marking on the site.

Each Project in the site is threated as a Conversation. The Conversation management can be found in the Dashboard menu with listed all chat panels and the option to review/edit/delete projects from there. Creating projects can be accessed in the Create Project menu in the navigation bar.

Each project is a complex item which should include the following content:

- Project Title – A short text that contains the name of the project.

- Description – Wider information about the task which can be viewed from view project button in the chat panel heading.

- Admins – This section should contain those users who will have the rights to edit the particular project. Users are divided in three groups: Front-enders, Back-enders and Full stackers.

- Developers –This is the section where are listed all developers that only have the right to view projects and chatting with other members without editing and deleting any of the projects. Again users are divided in the same groups as mentioned above.

- Chat history – every chat panel has his own history with all the messages sent from the creation of the conversation until real time and it can be read in the panel sliding up and down.

- Key words bindings – These buttons in the chat panel footer add different key words to the messages in order to alert particular group of users for some important tasks by the real time and email notifications.

# Modules

Quick description of the main project module.

* projectManager – This is the main angular module which includes all other little modules and initializes a default route for home screen view of the application. The home screen of the application is a login / register form separated one from another with tabs.
* projectManager.authentication – this module includes the service and the controller that are designed to operate with the login and register logic.

The register method includes the following fields:

* username: the name which will be shown as the user takes place in conversations
* email: existing email which is needed for the login method and which will receive email notifications
* password: standard password with length requirement of minimum 6 symbols
* confirm password: checks if the passwords match with the first one
* position: a group of radio buttons that defines what is the role of the user: Frontend , Backend or Full stack

The login method includes the following fields :

* email
* password
* projectManager.dashboard – this is a module that includes services and a controller for the projects and shows all the projects where a logged user takes place in the view as well as a filter for way that projects are represented in the dashboard view
* projectManager.editProfile – this is a module that allows the logged in user to change his/her profile information like username, email, password and position.
* projectManager.createProject – this module allows every user to create a conversation for a particular project; every project needs title, description and participants (admins and developers) to be created
* projectManager.editProject – a module that allows admins to update projects by title, description and participants

# Features

The main features of the project involve creating projects and participating in conversations. All the described above modules are integral part to the design of the site. The best feature is the chat participating and real time communication between developers and project managers. It involves using various AngularJS modules for both creation and the display of the given project conversation. The project also gives opportunity for email notification while particular users are offline.

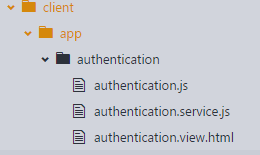
# Views

As the project is implemented with a ng-bp angular boilerplate the internal logic is handled with the given modules and their settings. The templates that are being used are always in the particular module folder of the application:

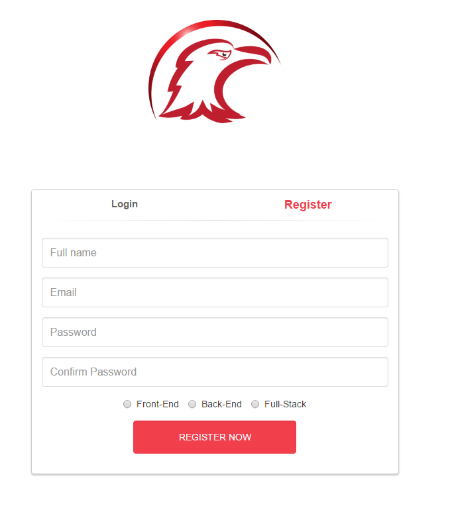
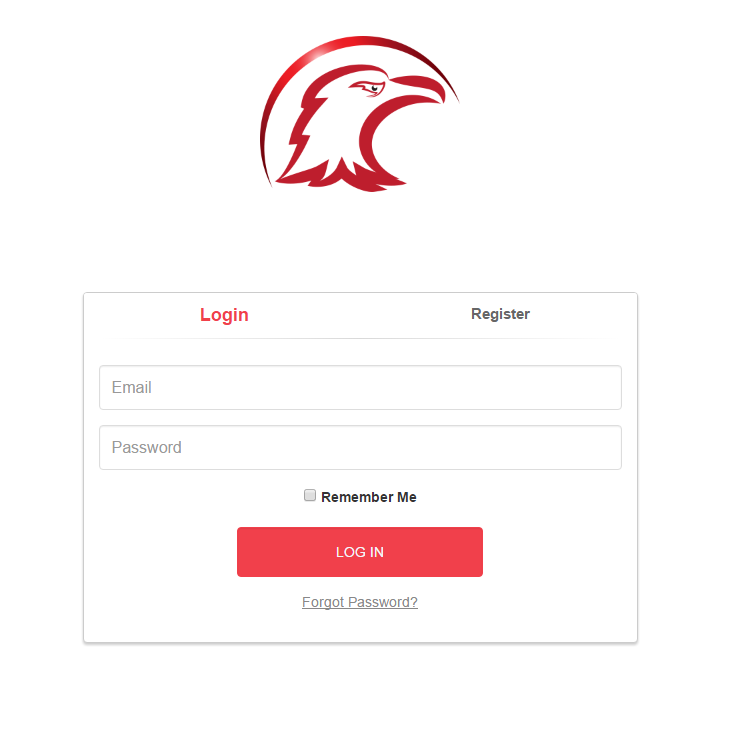
/client /app/<module-name> /

Explanation of the folders and their content will be done bellow:

[/client/app/authentication/authentication.view.html]

****

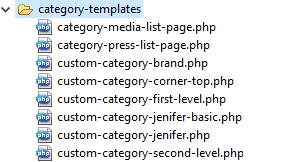
This folder, except of javascript controller and service for this module, contains the authentication view which is the default home screen of the application and it includes both login and register client side view separated by tabs as presented below:

****

The authentication view is set to default screen and it can be accessed with the default application URL address.

**Important !!! Based on the selected article type either the “Slides” functionality or the “Outfit Gallery” is used. The first three types of visual display use the “Slides” showing them on top, left or right of the article when displaying in the frontend. The last display type – Look Planche shows the “Outfit gallery” instead of the slides.**

[category-templates]



This folder contains the templates for category listing

The logic for which template is applied to which category has been implemented in the main folder of the theme inside the file **category.php**

[css]

This folder essentially contains all the CSS styles for the project

[feeds]

This folder contains 2 files which are the templates for the 2 RSS feeds available – RSS and RSS2

[images]

This folder contains all the statically available images (not user-uploaded) that are required by the design.

[inc]

This folder contains an empty file (central-content-slider.php) and 3 files which define

custom-header.php – custom header functions for the current template that will be displayed in Appearance> header admin panel

recent-posts.php – custom funtions for search and display of popular by category / popular posts.

Simple-content-slider.phptm – this file seems to be unused as it has some very generic JS attached to an event but the javascript is filled with dummy data.

[js]

This folder includes the scripts for the site. Most of the scripts are included by various modules. The custom ones are

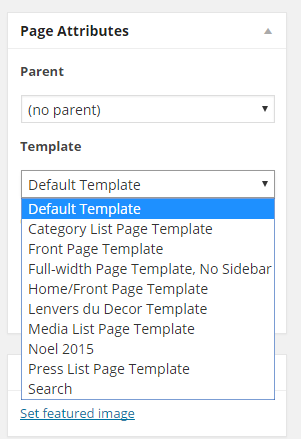
onload.js – a script to tell the site what ot happen on loading

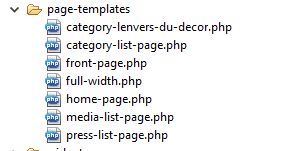
nagivation.js – a script to do the site navigation functionality.

AdminScripts.js – scripts to be executed in the admin panel to override title lengt validations.

[page-templates]

The page templates folder includes most of the templates that can be applied to the different pages of the site. The ones missing are Noel 2015 and Search which can be found in the main folder

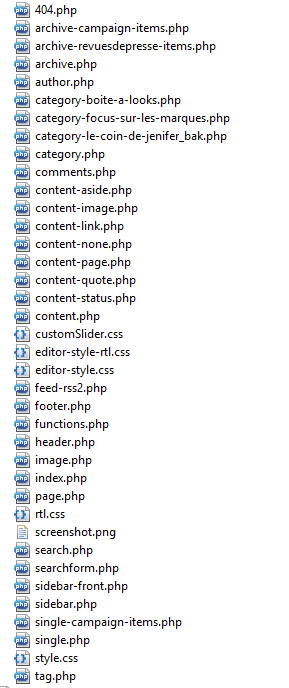




[widget]

This folder contains 2 scripts that are responsible for the generation of popular articles widget and the tags widget.

[main folder]

****

404.php – the template for page not found error

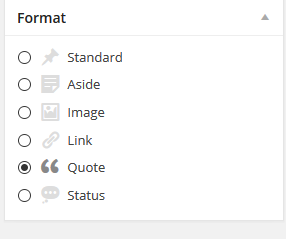
archive\* pages to show archived posts

author.php – template to show list of posts by author

category-\* templates for category display

category.php – logic for applying the correct template to each category listing

content\* - formatting of the post that can be found under Edit Post > format



footer.php – the ending part of the site. It requires the footer via the get\_ESI\_data function that is defined in functions.php. The file also adds a lot of custom javascript code for controlling the site – players, custom modifications to the themes etc.

header.php – the starting part of the site. It also requests the header from the main site via the get\_ESI\_data function also adding custom code for the lh-magazine

The rest of the files are named accordingly to what they do. Exception is the search.php template which is here instead of the page-templates folder, but is applied in that menu.

The functions.php file is as always the main place where custom code is being written. It contains ~ 2000 lines of different functions. Some of them are very trivial to just get a different format of the data. Some of the functions are making complex searches with mySQL directly. Here can be found the get\_ESI\_data function which does a curl request to fetch data from the main site.

# Project specifics

The previous section describes most of the specifics. In general all the custom code can be traced to functions.php file within the theme. The configuration file is located at the normal wp-config.php within the root of the site.

Important for the configuration is to set the curl addresses for the header and footer fetching.

define(**'**WP\_ESI\_URL\_HEAD**'**, **'**http://www.lahalle.com/catalog/product/gethead**'**)**;**

define(**'**WP\_ESI\_URL\_HEADER**'**, **'**http://www.lahalle.com/catalog/product/getheader**'**)**;**

define(**'**WP\_ESI\_URL\_FOOTER**'**, **'**http://www.lahalle.com/catalog/product/getfooter**'**)**;**

Another specific is that the site has 3 different setups for environment – local / development / production which are also set inside the config file. The site Root URLs and also the google analytics code is se there.

# Rights and users

The application has 3 levels of users for every project:

* Owner (project manager) – allowed to delete, edit and review projects
* Admin – allowed to edit and review projects
* Developer – allowed only to review projects

# Delivery

Describe if necessary specifics for delivery, something particular to know ? Specific manipulations ? Commands ?

*The system admin should be involved in the drafting of this part.*

# Development instance

If specific, describing the process of creating a local instance.

Describe if necessary prerequisites to an instance of development: settings.php, symbolic links, files, vhost, .htaccess, local configuration server ...

*The system admin should be involved in the drafting of this part.*