TEST PROJECT

BRIEF FOR TEST PROJECT FOR GLADEYE DEVELOPER APPLICANTS

Date: 30th May, 2016
Document Version: 1.0.4

PROJECT DETAILS

Client: Gladeye

Title: Test Project

Key Contacts Gladeye:

Michael Andrew
Systems Architect
michael@gladeye.co.nz

Ken Vu
Technical Director
ken@gladeye.co.nz

PROJECT SUMMARY

This project aims to provide a summary of relevant skills to Gladeye for applicants applying for developer roles.

It will test the applicant's ability to perform the following:

- Basic user account registration/management
- Facebook connectivity and authentication via OAuth
- Programatically sending emails via an SMTP server (i.e. using Gmail server)
- Generating a basic REST endpoint to interface with a database, sending back data as JSON

BRIEF

SERVER-SIDE SYSTEM & CMS

Using PHP/JavaScript libraries, PHP/CSS frameworks of your choice build a PHP based CMS that performs the following functions and stores user data in a MySQL database. You will be required to use PHP 5.6.x and MySQL 5.5.x for this project.

1. **User Management**

A basic user management system must be created to allow admins to perform the following functions:

1.1 **Create user accounts**

1.1.1 Users can be added by an administrator account and will have the following fields associated with them:

- First Name
- Last Name
- Email (valid email addresses must not match an existing user in the database)
- Password (passwords must be 8 characters or longer and contain a minimum of 2 numbers)
- Confirm Password (must match the Password field)
- Group (a option list consisting of the following options: User, Admin. Only a single group can be selected)
- Active (a checkbox which when unchecked will disable the user from logging in and will show an appropriate error message)
- Created date (hidden - a database only column that is updated when the user is created)
- Updated date (hidden - a database only column that is updated when the user is updated)
- ID (hidden - a database only column which is the unique ID of each user - this should be an integer)

1.1.2 All form fields are required and must be validated before the user can submit the form. A validation error message must be shown to the user if the form data is invalid.

1.1.3 When the user is created, they should be sent a welcome email stating that their account has been registered. See **3.3** for more information.

1.2 **Edit user accounts**

1.2.1 The fields described in 1.1.1 must also be editable after the user is created. The following additional read-only fields must also be displayed:

- Facebook ID

1.2.2 All form fields are required and must be validated before the user can submit the form. A validation error message must be shown to the user if the form data is invalid.

1.3 **List user accounts**

1.3.1 A page must be created to list all the user accounts. It must contain at least the following information on each user and must order the users by their first names:

- First Name
- Last Name
- Email
- Active
- Facebook ID
- Created Date

1.3.2 Each user must be able to be deleted from this list via a button, however the admin must confirm that they wish to continue with the deletion before the actual deletion is carried out.

2. Groups and Permissions

2.1 **Groups**

2.1.1 Two groups must be created that users can belong to: Admins and Users. A user can only belong to one of these groups at any given time. The data for these groups is not required to be editable via the CMS (i.e. the group name)

2.1.2 All users must belong to the Users group by default when their accounts are created.

2.2 **Permissions**

2.2.1 Only admins should have permission to access the **Add User** and **Edit User** pages of the CMS. Regular users may access the **List Users** page, but they cannot edit or delete users on the list.

2.2.2 A login screen must be created and all CMS pages must require the user to be logged in before the page shows. The user must use their email address and password to login. If the user enters incorrect credentials they must be shown an error message stating that their credentials were incorrect.

3. **User Registration**

3.1 **Standard Registration**

3.1.1 New users can register their own account from a **Register** link on the login screen. They will be asked to provide information for the fields described in **1.1.1**.

3.2 **Facebook Registration**

3.2.1 Users can login using Facebook. After they have authenticated successfully using the Facebook popup the CMS will need to check if the CMS user account exists by using the Facebook user's Facebook ID. If so, the user will be logged in as that user. If not, a new CMS user account will be automatically created by the CMS using their First Name and Last Name from their Facebook profile.

3.2.2 See Facebook's documentation on working with Open Graph and OAuth: https://developers.facebook.com/docs/facebook-login/

3.3 **Welcome Email**

3.3.1 When the user is created, they should be sent a welcome email stating that their account has been registered.

3.3.2 A third party SMTP server should be used to send the welcome email. Using the Gmail SMTP server is a good idea for this. The following article has the settings for Gmail's SMTP server: http://email.about.com/od/accessinggmail/f/Gmail_SMTP_Settings.htm

4. **CMS API**

4.1 **REST Service**

A basic REST service should be created to allow a list of user data to be retrieved via an HTTP GET request to the CMS.

4.1.1 An HTTP POST request to an authentication method (URL) should be made before further requests. The following POST variables should be included and should be authenticated against an admin account in the CMS: **email** and **password**. If authentication is successful, the following data should be returned as properties of a JSON object:

- token (a unique code as a string which can be used to make calls to other methods in the REST service
- success (a boolean value - TRUE if authentication succeeded or FALSE if failed)
- message (a success or failure message as a string)

4.1.2 A method should be created that allows HTTP GET requests to be made and will send back a JSON object which is an array of all the users in the CMS. Each user should contain the fields described in **1.1.1**. The correct headers should be sent back in the response to describe the response content type as **application/json.**

Q.A. / DEPLOYMENT

5. **Testing and QA**

5.1 The CMS should work on the following browsers:

- Internet Explorer 11+
- Firefox (latest version)
- Chrome (latest version)
- Safari (latest version)

5.2 The CMS should work on the following mobile OS:

- iOS 9+
- Android 4.4+

5.3 The source code for the project should be committed to a repository you will set up at github.com.