(Somewhat) Non-Technical Documentation

***Getting Started***

**Overview**

This application is a webapp content management system used to send out questions and then record the responses, of course with a lot more details in the actual code. It used the CRUD (create read update delete) model to manage the database entries and perform tasks as designated by the user.

**Installation**

The simplest form of installation is to go to [www.railsinstaller.org](http://www.railsinstaller.org) and follow the guide to installing rails on that website. (The project was made with Rails 3.2.13) It will guide you through getting the ruby and rails components and provide a brief introduction to the rails console.

If on Mac or Linux, the hands down favorite system is using RVM (Ruby Version Manager) to install and configure Rails and a program like Textmate to edit it.I would still recommend Netbeans if possible though. Netbeans is great because it will allows you to launch the program from the gui. Using other systems will require terminal commands to launch the server locally.

**Opening The Project**

Once Rails is installed, the next step is to choose how you want to open the project. We chose Netbeans as our IDE. However, you must use a version equal to or prior to 7.1. This is because future versions of Netbeans lost compatability with Rails. (If it does become available in the future feel free to use it). From there install the JRuby plugin and after that you should be able to open the project up in Netbeans.

The first think you have to do after opening up the project is to install the gems (the rails plugins used in our project). You can do this simply by opening up command prompt and navigating to the application directory. From here type the following command:

>> bundle install

You should see all the gems installed. Next step is to run the project which can be done by entering the command:

>> rails server

The app should open up in your default browser from the login screen. To make changes, simply edit the code and refresh the webpage. If you get an error “server already running”, navigate to tmp/pids folder in the application directory and and delete the file server.pid. You should now be up and running to manipulate the code and proceed forward with the project.

For more information, see the Rails README.rdoc file in the project directory. This explains the basics of how to use Ruby on Rails.

***Updating Content***

**Website Structure**

The website in structured in a way that each level of the file system is contained in a higher level. There the overarching layout which contains the main navigation menu; then inside that, depending on what is selected, the admins/assignments/questions/users will be contained. Then each of these containers will further contain views that represent the actions possible. Thus, say, if you want to edit the *delete* page in *users*, you do not have to worry about overarching features of the webpage such as body background. That is handled in the layout along with many other global features.

**Editing Individual Pages**

All the *html* code is located in the *app/views* folder. To edit the login page you must go into the access folder and edit *login.html.erb* . The code in this is mainly HTML. The *.erb* extension serves mainly to have the option of adding in ruby code so the compiler knows how to change it to the corresponding HTML when loading the page. Next, to edit the menu that shows up on all pages the file to edit is *views/layouts/admin.html.erb*. Finally, to edit the pages which show the actions related to the main menu options, you just have to go into their folders in the *views* folder and select the corresponding action. For example, if we want to edit the page which displays the deletion of a user, we would navigate to *views/users/delete.html.erb* .

**Changing Appearance**

Currently, the project uses the *Bootstrap* CSS/JS package put out by *Twitter*. This CSS and other future styling pages you want to add are in the *app/assets/stylesheets* folder. The nice thing about the compartmentalization of the webpages as described above is that we do not need to redundantly specify files. On the *Edit Admins* page, we need not include the styling for the general background of the webpage, since that is already referenced and in the layout file. Thus only put stylesheet links pertinent to the current part of the webpage you are working on. Whenever you want to link to a new stylesheet, first put that stylesheet in the *assets* directory and then add this line in the *head* of the HTML page:

<%= stylesheet\_link\_tag "your stylesheet name", :media => "all" %>

The rails application already knows the location of the stylesheet, so there is no need to specify the entire path.

**Adding Javascript**

Javascript can be added in-line in the normal HTML way. However, as this is often not considered best practice, we suggest you put all the required Javascript in the *app/assets/javascripts* directory. Then, similarly to CSS, and the link to the Javascript file to the *head*:

<%= javascript\_include\_tag "your JS file name" %>

For any other custom Javascript files simply just put it into the *assets/javascripts* folder and add the appropriate link.

**Configuration Files**

Since rails runs on a database, depending on the host database you will have to change the username and password to access this database. The place to do this is in the *app/config/database.yml* file. Enter the username/password of the database you’re using wherever you see the corresponding fields in this file.