Resume Critic - Documentation

David Mulatti, Kyle Petrozzi

1 Purpose

The purpose of the website is to be able to upload your resume and have it critiqued and be able to critique other resumes with comments. Once you read comments, you are able to read them and reupload/update your resume. You must create an account to post comments and upload your resume in order keep track of each comment and resume.

2 Installation

To install the website, simply extract the source code into your *public_html* folder. Then, replace the values in *dbaccess.php* with the appropriate information for the database you'd like to use. Lastly, navigate to "/admin/create_db.php" to create the database, then navigate to "/admin/populate_db.php" to populate the database with four test users, and an admin account.

Login Info

User Name	Password
admin	password
testone	password
testtwo	password
testthree	password
testfour	password

For a working demo, this site is live at 334.mulatti.com.

3 Database Design

The database for this site contains three tables: users, comments, and ratings. The users table stores user info, including uwinid, hashed password, full name, rating on their resume, resume upload date, description for their resume, and a boolean indicating whether they have uploaded a resume or not.

The comments table stores an auto incrementing comment id, the uwinid of the commenter, the uwinid of the user the comment is for, the date of the comment, and the comment itself.

The ratings table is almost identical to the comments table, but does not store the date, and is used to store ratings represented as a double rather than comments.

4 Site Walkthrough

4.1 Design and CSS

Bootstrap is used for the design of this site, with the assistance of a kickoff template provided by maxdesign [1]. There are modifications made to this, located in /assets/css/styles.css.

4.2 Admin Control Panel

When logged in as admin, an "Admin" option appears in the site header that will allow one to access the admin control panel. In this, the admin can view all of the tables in the database, edit any user, delete any comment, drop the database, create the database and populate the database with the test data shown above.

Also, when logged in as admin, the edit user page is extended to include more options, and allows the admin to edit any user they wish.

4.3 Header

The header is included within every page that is viewed by the user. It contains most of the sites navigation features and allows the users to easily navigate the website. It is dynamic as it changes with who is logged in (such as the admin, admin pages will appear). It contains different links as well as the login form. The login button brings you to the login.php page. It uses the session variable "logged_in" to control most of the dynamic content.

4.4 Footer

The footer is included in every page, and is simply the closing ⟨body⟩ and ⟨html⟩ tags, as well as a centered message including an emoji. The emoji is generated using Twitter's *twemoji* plugin [2].

4.5 Create Account

Here, there is a is a web form that takes a uwinid, password, and full name to create a new account. There is also a captcha, provided using Google's reCAPTCHA api, to verify that the person signing up for an account is indeed human. The form uses the jQuery Validation plugin [3] to validate all the fields, including AJAX calls to the database to ensure that a username is not taken. It will also ensure that the password is at least 8 characters long, and that the full name only has alphanumeric characters in it.

On submission, all fields are sent as POST variables to *createaccount_go.php* which will insert a new user into the database. The password is hashed using the *password_hash()* function, provided by ircmaxwell [4].

4.6 Login and Logout

The login.php is relatively simple. It compares the password sent within the form to the hashed password stored in the database associated with the name sent in the form. If it matches it sets the *logged_in* session variable to 1, and then redirects back to the index page. It also checks if the user logging in is the admin, and if so, changes the session variable to 3. It also has an error page if the wrong username or password is entered.

4.7 Edit User

The edit user link appears in the header whenever a user is logged in. A user can then edit their Full Name or their password. If an admin is logged in, then the edit user page is extended, allowing the admin to change many more options for any user they wish.

4.8 Resume Upload

This page is used to upload/edit your own resume page. If you are logged in you can add a resume and description. If you already have a resume uploaded, it shows it to you, and auto fills the description with your last description used. When you use the upload button it calls the upload.php class. This class checks if the file uploaded is a application/pdf MIME type, if it comes from the POST function, and finally if it is not too large of a file. If all of these checks pass, then it saves the file as *uwinid*.pdf, *uwinid* being the username of the current user. It also updates the upload date, *has_uploaded*, and the description columns within the database to the user who uploaded. It also will add a comment to the resume, notifying that a new resume has been uploaded. It then redirects back to the resume viewer page. It also has error pages that tell the user what went wrong if there is a upload error.

4.9 Resume Viewer

When a name on the homepage is clicked, the user is brought to a page which displays the resume as an embedded object on the left, and the description, upload date, and comments on the resume on the right. Above the resume there is a current star rating of the resume, gathered by what the users have voted.

If the user is logged in, below the comments there is a textarea to add their own comment, and an option to rate the resume out of five stars. The star rating system is powered by a jQuery plugin for bootstrap made by GitHub user *kartik-v* [5].

4.10 Contact Us

The "Contact Us" page displays our names, some information about ourselves, and our email addresses. There is also a link to the GitHub page that our code is available on, and a form including a TinyMCE [6] editor to send an email directly to us using the PHP *mail()* function.

On submission of the contact form, the user is redirected to *contact_go.php* which takes the POST variables from the form, validates the captcha, and processes the message given from the TinyMCE editor through XSLT to strip out everything that isn't a ⟨b⟩, ⟨u⟩, or ⟨i⟩ tag.

References

[1] russmaxdesign, *Bootstrap Kickoff Template*
Available at: https://github.com/russmaxdesign/01-bootstrap-kickoff-template

[2] Twitter, *Twitter Emoji For Everyone*
Available at: https://github.com/twitter/twemoji

[3] jQuery Validation
Available at: http://jqueryvalidation.org/

[4] ircmaxwell, *password_compat*
Available at: https://github.com/ircmaxell/password_compat

[5] kartik-v, *bootstrap-star-rating*
Available at: https://github.com/kartik-v/bootstrap-star-rating

[6] TinyMCE
Available at: https://www.tinymce.com/