**griefSpace**

Brief introduction on features and the logic

1. **Journal Page**
2. **Login status validation**

Once users enter the journal page, the script will first check if the user has logged on by validating its Php session variable. Actions such as fetching journals will be executed if the variable is set. Otherwise, the journal page will redirect the user to the *loginPage.php* to ask to log in.

1. **User’s id and Journals fetching**

Once the session variable is confirmed to have values, scripts will fetch its related user’s id from the table userdata. It will then fetch all existing journals written by this user and populate the journal panel by rendering journal boxes based on the number of entries fetched.

1. **Content displayed**

Each journal box will display two pieces of data, the journal’s title and its creation timestamp. The creation timestamp is converted to a readable date format (i.e., YYYY-mm-dd) by using a Php *DateTime* object.

1. **Clickable journal box and redirection**

Users can click on each journal box to enter its specific preview panel. Details of the preview panel will be explained in the next section. Once users click on a journal box, scripts will fetch its journal’s id from the table *journal\_basic*. It will use the fetched id to construct the path of the preview panel by combining it with the file location.

1. **Preview Page**
   1. **Login status validation**

Once users enter the journal page, the script will first check if the user has logged on by validating its Php session variable. Actions such as fetching journals will be executed if the variable is set. Otherwise, the journal page will redirect the user to the *loginPage.php* to ask to log in.

* 1. **Session Variable and URL parameter**

Once users enter the preview page, scripts will process its session variable and URL parameter. The session variable should contain a username string, whereas the URL parameter should contain a value of the journal’s id.

* 1. **Fetching and populating**

Scripts will first fetch details of this journal from the table *journal\_basic* to retrieve its title, user’s emotion and questions’ ids. Questions’ ids will then be used to fetch questions’ context by communicating with the table *journal\_question*. Eventually, this fetching action will render numerous question boxes and populate the preview panel.

* 1. **Contents**

*Preview.php* allows users to view a journal's title, user's emotion and all fetched questions. Users can click on the desired question to enter the details page and review relevant answers. Users can also perform an editing action by clicking on the edit button.

1. **Clickable journal box and redirection**

Users can click on each question box to enter its specific details page. On the details page, users can then review all relevant answers related to this question. Explanations of how the details page work will be illustrated in the next section. Once users click on a question box, scripts will fetch its unique question’s id from the table *journal\_question*. It will use this fetched id to construct the path of the preview panel by combining it with the file location and the journal’s id.

1. **Details Page**
   1. **Login status validation**

Once users enter the journal page, the script will first check if the user has logged on by validating its Php session variable. Actions such as fetching journals will be executed if the variable is set. Otherwise, the journal page will redirect the user to the *loginPage.php* to ask to log in.

* 1. **Session Variable and URL parameter**

Once users enter the details page, scripts process its session variable and URL parameter. The session variable should contain a username string, whereas the URL parameter should contain the journal’s id and question’s id.

* 1. **Fetching and populating**

Scripts will use the journal’s id, user’s id and question’s id as conditions when it performs the fetching request. The preview panel will then be populated with the journal’s title, the user’s emotion and all answers to a selected question.

1. **Creating Journal**
   1. **Setting title and emotion**

On the first page of the modal page, there are two primary containers. Users can first give a title by typing in the user’s input.

Moreover, users can select one of the five emotions from a container. Each emotion is presented by an image and a description. Once the user chooses an emotion, the selected emotion and its element will be modified to have a different size from others.

Moreover, the input event listener validates the length of the title input. If the title’s length is greater than 0, it will give a true value. On the other hand, the selection of emotion is validated by the click event listener. It will give a true value if one emotion is selected. Only if both title input and emotion selection give true values will it enable the ‘next’ button.

* 1. **Selecting and answering questions**

A question is set by default to prevent the question from displaying nothing to the user. However, users can always click on the question to manually select a preferred question from the question pool. Each question is pre-defined and adapted from Neimeyer’s book Techniques of Grief Therapy.

Users can answer a question by entering their inputs in a text area. An input event listener validates the length of the answer input. It will give a true result and unlock the ‘add question’ button if the length of the answer input is greater than 0.

* 1. **Adding questions**

Users can also answer more than one question by pressing the ‘add question’ button at the bottom of the page. Once the button is pressed, an HTML element will be created, set to have some stylings and appended to the page. Once the new element is ready, it will enter the window, whereas the previous question will leave the window. However, users can always edit any previous sections if they want.

* 1. **Editing previous sections**

Users can click on the ‘before’ button to navigate to the previous container and edit its details. Users can also click on the title or emotion directly to navigate to the page where users can update the title and change emotions.

* 1. **Save and Warning**

Users can click the ‘save and exit’ button to insert the data into the table *journal*\_basic. Once the request is submitted, scripts will verify the title’s input, the emotion’s selection, questions texts and answer inputs. If any of the validation returns false, the request will be rejected. Users will then be notified by a pop-up window which indicates the errors. For example, if the error results from an empty text area, it will bring the user to its container. Scripts will also check if the users exceed the limit of creating a journal before executing the ‘insert’ statement.

* 1. **Time limit**

Users can create up to one journal every day.

* 1. **Alert email on journal created**

If the request is successful, the script will first fetch a boolean value from the table user\_notification. If the column of ‘*journal*\_*created*’ contains a true value, users have agreed to receive an alert on the journal created, and an email will be sent via email.

1. **Editing Journal**
   1. **Editing context**

The editing system allows users to edit the journal’s title, emotions, questions and answers after creation. It also allows users to add more questions to existing journals by clicking on the ‘add question’ buttons.

* 1. **Similar validation system with ‘Creating journal’ feature**

The editing system has the same validation checks on emotions and title inputs. It also has the same validation function on the question and answer inputs.

* 1. **Edit with no time limit**

However, unlike the ‘creating journal’ feature, this feature allows users to update the journal’s data without any limit on times and timestamps.

* 1. **Alert email on journal updated**

If the request is successful, the script will first fetch a boolean value from the table user\_notification. If the column of ‘*journal\_updated*’ contains a true value, users have agreed to receive an alert on the journal update, and an email will be sent via email.

1. **Insight Panel**
   1. **Data fetching**

After users enter this ‘insight’ page, scripts will fetch a string of emotions and timestamps from the database. The emotion was tagged and selected by the user when creating the journal, whereas the timestamp represents the creation time of a journal in epoch format.

Strings of emotion will form an array, whereas each of them will later be searched and replaced by a hex colour code.

Different epoch timestamps will form an array, whereas each element will be converted to a date format (i.e., YYYY-MM-DD). This specific format is used as it matches the date format used by fullCalendar.io.

* 1. **Populating day cells**

A day cell will be populated by a string of emotions and set to use a background colour if the date it represents matches the timestamp.

* 1. **Day cell and linking to preview page**

Users can click on each day cell to get redirected to the preview page to review a matched journal.

Once the user clicks on the day cell, it will call a function to retrieve its date string and parse it to a Php DateTime object. This date string will then be converted to an Epoch timestamp and later used to compare with timestamps stored in the database.

For example, if the date string retried from the calendar is 2022-08-05, the DateTime object will generate an Epoch timestamp representing the **start** of 2022-08-05. Another DateTime object will then be created and generate an Epoch timestamp representing the **end** of 2022-08-05.

Comparing those timestamps allows finding the matched journal and its id. The journal’s id will then be used to construct the path to the preview page.

1. **Setting Panel**
   1. **Support changing email and password**

Users can click on the ‘Account’ div and view their usernames and email addresses on the account panel. The account panel also supports users to change their email addresses or passwords by clicking on the buttons.

* 1. **Customisable notification settings**

Once users enter the ‘push notification’ section, the application will fetch settings from the table *user\_notifcation*. Results will be processed and used to indicate that this setting has already been turned on. Users can update their preferences by clicking on the switches. Once a switch is clicked, an updating request will be processed by a JavaScript AJAX call to update the relevant value on the database.

Notification settings now support alerting users if any of the following events happens:

* Email updated
* Password updated
* Account activated
* Journal created
* Journal updated
* Login activity recorded

1. **Login and Registration System**
   1. **Registration and activation**

A user must provide a username, email address and password before submitting a registration request. Once the user submits the form, scripts will validate the user's inputs to implement several rules to ensure inputs are valid. The password will also be checked based on the rules as the password should be at least 8-digit long and include at least one uppercase letter, one lowercase letter and one symbol. If the username and email do not exist in the database, the user's information will be inserted into the *userdata* table.

Moreover, scripts will hash a generated binary token to generate a hyperlink by parsing it as a URL parameter. Scripts will then send an activation email with this hyperlink to inform the user of the registration activity and ask to finish the registration procedure.

Once the user clicks on the hyperlink, it will redirect the user to an activation form which requires him to enter their password to confirm his identity. The algorithm will also ensure the token is matched and the request has not expired.

This activation system aims to improve security by confirming user identity. The email can also alert the user of any suspicious if the user is not the one who submits the request.

* 1. **Resetting password system**

This system allows users to update their passwords if they forget their passwords. Users will need to request a resetting password email to obtain a token and a hyperlink via email. Once users gain access to the form and enter the required data, scripts will validate the token and access time to ensure the request is valid.

* 1. **Login system, OTP and its alert email**

Users can log in by accessing loginPage.php. If the user’s inputs are validated and their identity is confirmed, scripts will set a Php global variable $\_SESSION to allow other pages to communicate its values.

If the user has turned on the one-time password option on the settings page, an email containing a generated 6-digit one-time password will be sent to the user’s email inbox. The user will need to enter this passcode to perform a two-factor authentication and confirm his identity.

If the user has agreed to receive a login alert, an email will be sent to the user to inform of this login activity. This alert aims to alert the users and reduce the response time to any identity thefts.