**DESIGNING**

CSBEES

:: The Open Source Notes Sharing Application ::

[**http://www.csbees.neurals.in**](http://www.csbees.neurals.in/)[**https://github.com:/hay-wire/csbees**](https://github.com:/hay-wire/csbees)

VISUAL INTERFACES

1. **Home**

After a user is logged in, he/she may visit the home page using top navigation bar. Home page contains the list of those notes which have been uploaded under a tag to which the user has subscribed.

The list is in reverse chronological order, recent most note is displayed at the top.

1. **Upload Notes**

**New Notes Form:**

This is the interface for uploading a new note.

**Title:** is the title which is shown as the heading of the note. It should be short and to the point.

**Description:** is the description of the note. It contains a short write up of the contents of the notes being uploaded and credits, if any.

**File:** this button is used to navigate and choose the notes file which is to be uploaded.

**Tags:** These are the short categorizers which help other users to identify the relevance of the note, its domain and category. User starts typing and matching tags are shown below so that the user may choose one of them. Multiple tags are preferable for a note. This way it becomes accessible to many users having interest into that type of notes.

**New Tags Form:**

**Tag’s Full Name:** It is the full name of the university, or college or subject, etc.

**Tag’s Short Name:** It is the acronym of the tag which is displayed to the user

**Tag’s Category:** It is either a college/university, subject or a popular topic. If the user creates a tag for which whether the acronym or the full name already exists in the database, It is shown to the user before creating new tag itself

The user is first checked if he is logged in or not. If not, he is sent to the login page.

Now, when a user uploads a new note on the server, all the details are properly escaped so as to avoid any SQL injections. The submitted notes file is also verified for invalid extension. If all things are ok, the file is renamed and moved to “uploads folder” and an entry with all the given details are made into the database. The file becomes available to all users as soon as it is uploaded and verified by the system.

1. **My Uploads**

This is the interface for showing those notes which a user has uploaded. It selects all those notes from the database where the uploader is the current user logged in and displays it.

1. **All Notes**

This is the interface for showing all the notes available on the server. It selects all the notes which are present in validUploadsView in the database.

1. **Profile**

This interface has two tabs. One is for selecting a name and other for subscribing to available tags.

A user can choose any nickname not choosen by anybody else. It will be shown against his/her uploads.

A user can choose as many tags as he/she wants to be displayed in his home page. These tags’ id are stored against a user’s user id in “usersTags” table and fetched via “usersTagsView”.

FILES ARCHITECTURE

1. **/includes.php**

This file is responsible for configuring the application instance each time a user connects to the website. It defines vital constants used throughout the application and holds the important environment configurations such as error\_reporting, etc. You need to change the details when a new instance of the application is installed on a system.

1. **/AuthManager.class.php**

This class is responsible for new signups and logins. It manages how a user is logged in via Facebook. Once the user is authenticated, his details are brought from the database and given to he SocialUser class.

It also handles Facebook posts for new user signup announcement and new upload announcement over Facebook account of the user

1. **/CSBeesManager.class.php**

This class is responsible CSBees specific operations such as new notes uploading, handling voting on the notes, managing tags related to a note, etc.

1. **/download.php**

This file is responsible for getting the download link from the database and updating it for download count.

1. **/DB.class.php**

This class is responsible for user related database actions such as managing supscription tags, updating user details, adding user when a new user sign up, getting the user details if a user logs in, etc.

1. **/functions.php**

This file is a collection of common functions used throughout the application such as generating page numbers when so many files are to be displayed, sanitizing user inputs, triggering website events, etc

1. **/index.php**

This is the bootstrapping file of the application. It manages everything right from the beginning of a new request to sending the response.

It is responsible for calling includes.php and instantiating AuthManager and CSBeesManager classes to keep a check on security issues.

It then checks for any user input, if present, processes it and gets the output in the buffer.

After that it checks which page should be loaded next and calls the corresponding view dynamically.

Finally the output is flushed to the user.

1. **/login.php**

This file is comes into action when either a user login fails or a user is returned after being authenticated by Facebook.

If the login has failed, it provides a link to relogin. Otherwise, the user is redirected to his home page.

1. **/logout.php**

This file is responsible for terminating user session and deleting the cookie set by Facebook so that the user is logged out of the application.

Thereafter, the user is sent back to the index page.

1. **/SocialUser.class.php**

This class is the user details holder. It stores all the user related details such as user id, name, ip, lastActivityTime, etc. It has methods to provide these details to other controllers also in case needed. An instance of this class is always loaded into the session.

1. **/fb/SocialFB.class.php**

This class is the wrapper of Facebook-php-sdk and provides common mechanisms to authenticate user, get user details, and start a session for a Facebook user on CSBees. It uses Facebook-php-sdk to get the details from Facebook’s server.

1. **/ajaxRequests/ajaxTagsCreator.js**

This file is responsible for asynchronously sending tag creation requests to the server after taking the input from the user and displaying the result.

1. **/ajaxRequests/ajaxVoteup.js**

This file handles vote up and vote down requests on the client side and sends the details via ajax to the server in proper format.

1. **/ajaxRequests/newTagsHandler.php**

This file is responsible for accepting requests from the ajaxTagsCreator.js file asynchronously and creating or rejecting new tags creation requests.

Proper output is sent back.

1. **/ajaxRequests/returnTags.php**

This file is responsible for returning matching tags as a user types in words in the tags input form. It encodes data in JSON format