COMP3322 Modern Technologies on World Wide Web

Assignment 1: A Simple Webmail System (11%)

[Learning Outcomes 2, 3]

Due by: 23:55, Thursday October 19, 2017

Total mark is 100.

Overview

In this assignment, you are going to develop a simple **web-based email system** using PHP, JavaScript, AJAX, HTML and CSS. The webmail system implements a few simplified functionalities, including displaying emails in your mailboxes, deleting emails, moving emails from one mailbox to another.

Task 1 (15 marks) Create basic index.html and prepare database tables

Create **webmail.html** which renders the layout shown in the following figure, i.e., the page has four divisions: heading, function buttons, mailbox list and email list. You can design any heading, showing the name of your webmail system. In the function division, there should be four types of buttons: "Delete", "Move to xxx" (where "xxx" is decided by which view the page is currently in, to be detailed in Task 2), "<" and ">". You can implement the buttons in any fashion you like, as long as they are clickable and upon clicking, the corresponding functionality can be achieved. In the mailbox division, include three links: "Inbox", "Important", "Trash". In the email division, list each email in the format given in the figure (checkbox, sender, title, date).

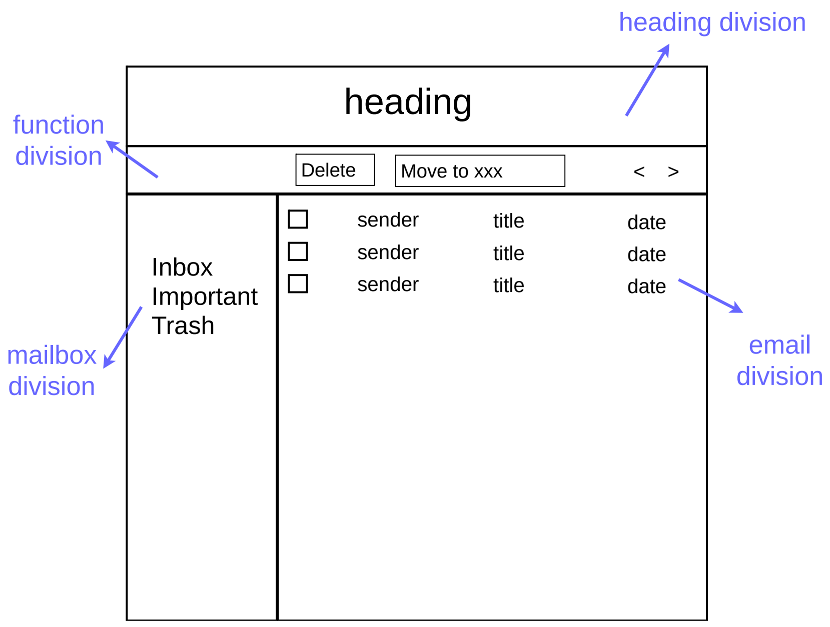

Fig. 1

Prepare the database on the server side:

1. Create a database table **emails** in your database in the server **sophia.cs.hku.hk**.

Each email entry in the table must contain the following fields:

- **emailID** – The unique ID of the entry. Assume that the emailID is an integer starting from 1.
- **sender** – The email address of the sender (less than or equal to 20 characters).
- **title** – The title of the email (less than or equal to 30 characters).
- **date** – The date when the email is sent (less than or equal to 20 characters).
- **content** – The content of the email.
- **mailbox** – The mailbox where the email belongs to (less than or equal to 10 characters).

For your reference, the following SQL creates the email table.

```
CREATE TABLE emails (
    emailID int(11) NOT NULL,
    sender varchar(20) NOT NULL,
    title varchar(30) NOT NULL,
    date varchar(20) NOT NULL,
    content longtext NOT NULL,
    mailbox varchar(10) NOT NULL,
    PRIMARY KEY    (emailID)
 )
```

The following SQL inserts an email entry into the emails table.

```
INSERT INTO emails(emailID , sender, title, date, content, mailbox)
VALUES (
1,
'Jim@cs.hku.hk',
'Hello from Toronto',
'Sep 25 2017',
'Hi Alice, I am traveling in Toronto now.',
'inbox'
);
```

Please prepare at least 3 pages of emails in the database table. That is to say, if you set the number of emails to display on each page in the email division to be 5, then you should prepare at least 15 email entries in the database table. Suppose the emails are stored in chronological order, i.e., the email entry with emailID=1 is the oldest email, and the email entry with emailID=2 is the second oldest email, etc. In this way, you can order the emails in chronological or reverse chronological order according to their emailIDs.

Task 2 (25 marks) Display the email list

When **webmail.html** is first displayed, it should be showing the “Inbox” view: The “Inbox” link in the mailbox division should be in a different color (e.g., “color 1”) from the color of the other mailbox links (e.g., “color 2”), the “Move to xxx” button should be stating “Move to Important”, and a fixed number of emails in the inbox are listed in the email division, following the format shown in Fig. 1 (you can specify the number of emails for display on a page).

After you click “Important” in the mailbox division, the web page should be showing the “Important” view: The “Important” link turns into color 1 and the other two links are in color 2, the “Move to xxx” button should be stating “Move to Inbox”, and a fixed number of emails in the “Important” mailbox will be listed in the email division (according to the number you specified).

When you click “Trash” in the mailbox division, the web page should be showing the “Trash” view: The “Trash” link turns into color 1 and the other two links are in color 2, the “Delete” button should state “Delete Forever” instead, there are two “Move to xxx” buttons (instead of one) stating “Move to Inbox” and “Move to Important” respectively, and a fixed number of emails in the “Trash” mailbox will be listed in the email division (according to the number you specified).

If there are more than one page of emails in the corresponding mailbox, emails on other pages are displayed when you click “<” or “>” in the function division.

Please note that switching the mailbox view, and clicking < or > should load the corresponding email list in the email division **without reloading the entire web page.**

1. JavaScript events and event handlers.

1.1 When **webmail.html** is first loaded, load your specified number of emails in the Inbox in the email division, and order them in reverse chronological order. (**Hint**: register a JS event **onload** in the **<body>** tag)

1.2 When a mailbox link in the mailbox division is clicked, load your specified number of the newest emails in the corresponding mailbox in the email division, order them in reverse chronological order, and display the corresponding function buttons in the function division. (**Hint**: use the JS event **onclick**).

1.3 Whenever “<” or “>” is clicked, newer or older emails in the corresponding mailbox (of your specified number) will be listed in the email division. (**Hint**: use the JS event **onclick**).

1.4 Implement AJAX codes in the event handler functions to communicate with the server for retrieval of emails from the database.

2. Server side processing logic.

Create a PHP file **handleEmailDisplay.php**, which should retrieve information of a number of emails in the respective mailbox from the emails table in the database, according to the request from the client, and send them to the client (note that only necessary information for the email list display should be retrieved and sent to the client, e.g., the email content is not retrieved/sent). For database connection and query, refer to pages 16-17 in lecture slides **5_PHP_II_COMP3322_f2017.pdf**.

Task 3 (20 marks) Display email content

When you click each email entry in an email list display (on sender, title or date), a request will be sent to the server, which will retrieve the content of the email from the emails table in the database, and send it back to the client. The client should display the content in the email division, replacing the email list (You can display the content of an email in any format you like), and disable the function buttons in the function division by making them unclickable. This should not cause reload of the entire web page as well.

Similar to Task 2, you should decide the JS event to trigger, implement the event handling AJAX code on the client side, and a PHP file **handleContentRequest.php** on the server side to receive your request and retrieve the corresponding email content from the emails table in the database.

Task 4 (30 marks) Delete and move emails

To the left of each email entry in an email list display, there is a checkbox, which you can click to check and uncheck. When the web page is displaying the “Inbox” view, after you select some emails and click the “Delete” or “Move to Important” button, a request will be sent to the server to update the “mailbox” field of the corresponding email entries to “trash” or “important” correspondingly.

When the web page is displaying the “Important” view, after you select some emails and click the “Delete” or “Move to Inbox” button, a request will be sent to the server to update the “mailbox” field of the corresponding email entries to “trash” or “inbox” correspondingly.

When the web page is displaying the “Trash” view, after you select some emails and click the “Move to Inbox” or “Move to Important” button, a request will be sent to the server to update the “mailbox” field of the corresponding email entries to “inbox” or “important” correspondingly; if you click “Delete Forever”, a request will be sent to the server to delete the corresponding email entries from the emails table in the database.

After these updates have been successfully executed on the server, the server will further retrieve information of Y immediately older emails (if you deleted or moved Y emails in the current email list display), and send them to the client. On the client side, you should remove the selected email entries from the email list display in the email division, and append the Y newly retrieved email entries to the bottom of the list, such that the total number of email entries displayed is still the same as before (or smaller only if all older emails have been retrieved). All these should not cause reload of the entire web page as well.

Similar to Task 2 and Task 3, you should decide the JS event to trigger, implement the event handling AJAX code on the client side, and a PHP file **handleMailboxChange.php** on the server side to receive your request and update database accordingly.

Task 5 (10 marks) Style the page using CSS

Please use a separate **style.css** file to include all your styling rules.

1. (**5 marks**) Use CSS styling you choose to make your page look nice with good layout
2. (**5 marks**) Implement Responsive Web Design to make your page look nice on screens of different sizes.

Notes:

1. You can use either basic JavaScript or jQuery to implement client side scripting.
2. Maintain a good programming style: avoid redundant code; the code should be easy to understand and maintain.
3. You should carefully test all the functionalities stated in this handout.

Submission

You should submit the following files:
(1) webmail.html
(2) style.css
(3) handleEmailDisplay.php
(4) handleContentRequest.php
(5) handleMailboxChange.php
(6) exported tables you created in your database (on phpMyAdmin, you can export your tables on the “Export” tab: choose “Quick - display only the minimal options”, Format: “SQL”, and click “Go”.)

Please compress all the above files in a .zip file and submit it on Moodle:
(1) Login Moodle.
(2) Find “Assignments” in the left column and click “Assignment 1”.
(3) Click “Add submission”, browse your .zip file and save it. Done.
(4) You will receive an automatic confirmation email, if the submission was successful.
(5) You can “Edit submission” to your already submitted file, but ONLY before the deadline.