

CHRISTOPHER NEWPORT UNIVERSITY
DEPARTMENT OF PHYSICS, COMPUTER SCIENCE & ENGINEERING
CPSC 360 - PROGRAMMING LANGUAGE CONCEPTS

*** Assignment 5: Words Galore! ***

Instructor: Dr. Roberto A. Flores

Goal

Develop a web site using PHP, JavaScript and MySQL.

Web Interface

You are developing *WordsGalore*, a web site where users can browse words alphabetically.

The figure below shows the portal's home page, which displays the information from the Home tab.

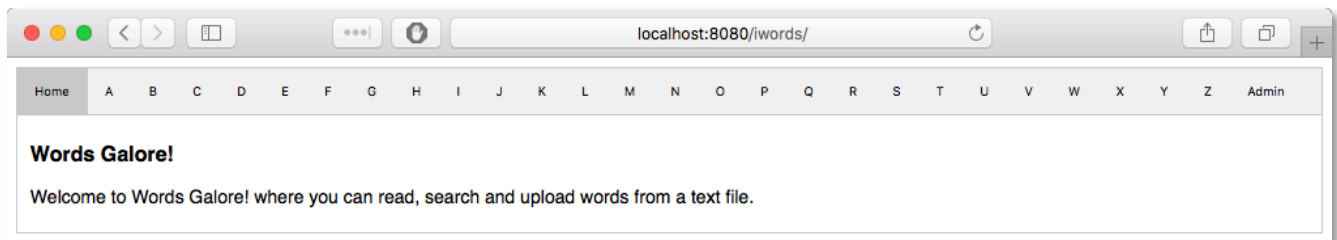


Figure 1. Web page showing the Home tab.

Subsequent tabs with letters A-to-Z display the words that begin with that letter. The figure below shows that choosing tab A lists all words that begin with letter A, which in this case are a total of 13,758. Other tabs display words that begin with their corresponding letter.

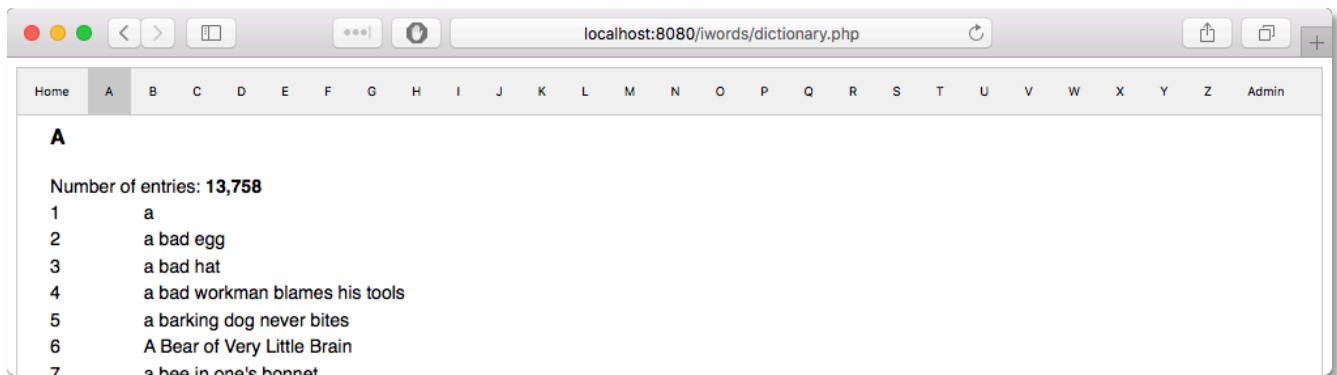


Figure 2. Web page showing the words on the A tab.

The Admin tab is used by system administrators to add words or delete all words from the web site. The page (below) requests a password to allow access.

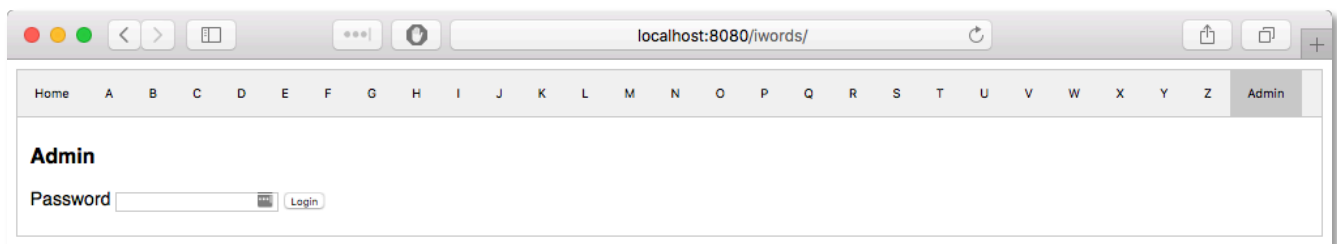


Figure 3. Web page showing the Admin tab.

If the Login button is pressed without providing a password, a JavaScript dialog (below) indicates that one must be provided (the figure shows a popup dialog the way they get displayed in Safari).

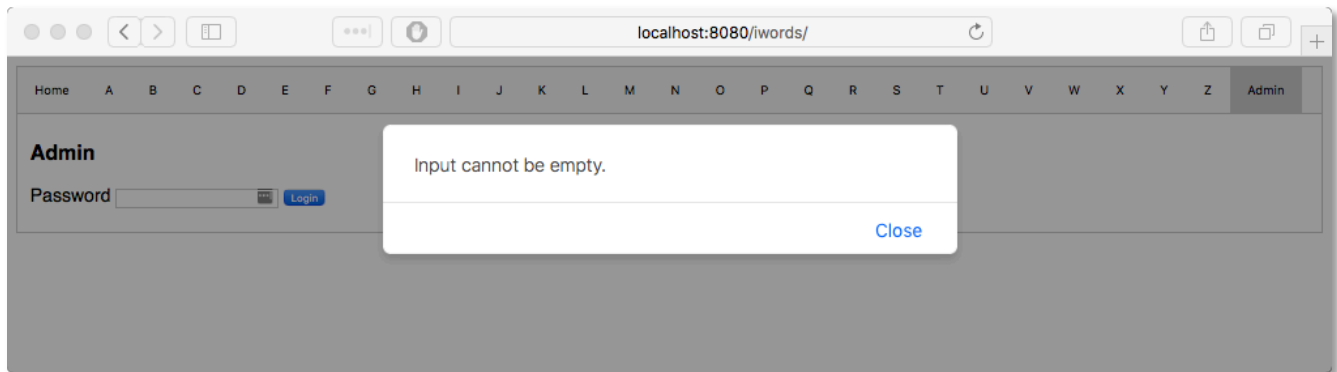


Figure 4. JavaScript dialog indicating that a password must be provided.

The page (below) displays a message if the password is incorrect. Pressing Try Again loads the home page.

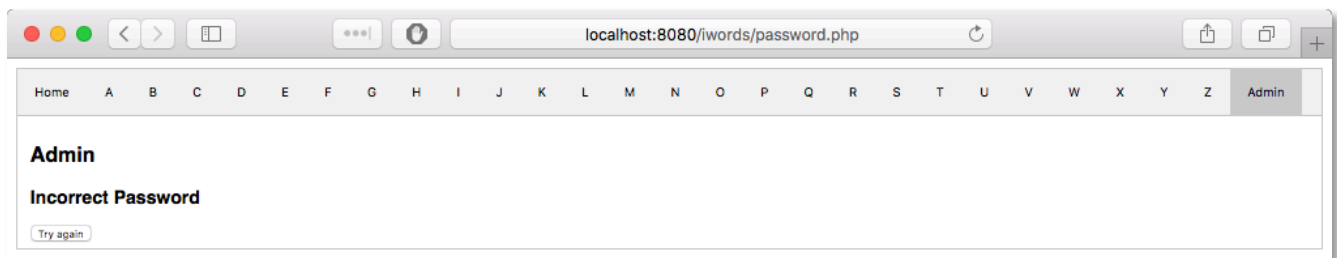


Figure 5. Web page indicating that the password provided was incorrect.

The administrator's web page (below) is displayed after providing the correct password. The page has three options: upload data, delete data, and logout. Pressing the Logout button would display the home page (so that subsequent attempts to click on the Admin tab will request a password, as shown in Figure 3).

Developer Note: The administrator password must not be hard-coded in your PHP files. Rather, the password must match the one assigned to the administrator on the local MySQL. To achieve this feat (as shown in lecture), write in your PHP code a command creating a database handler using the password given and the administrator's username (usernames are given in a later section). The command will succeed if and only if the password given matches the password assigned to the administrator.

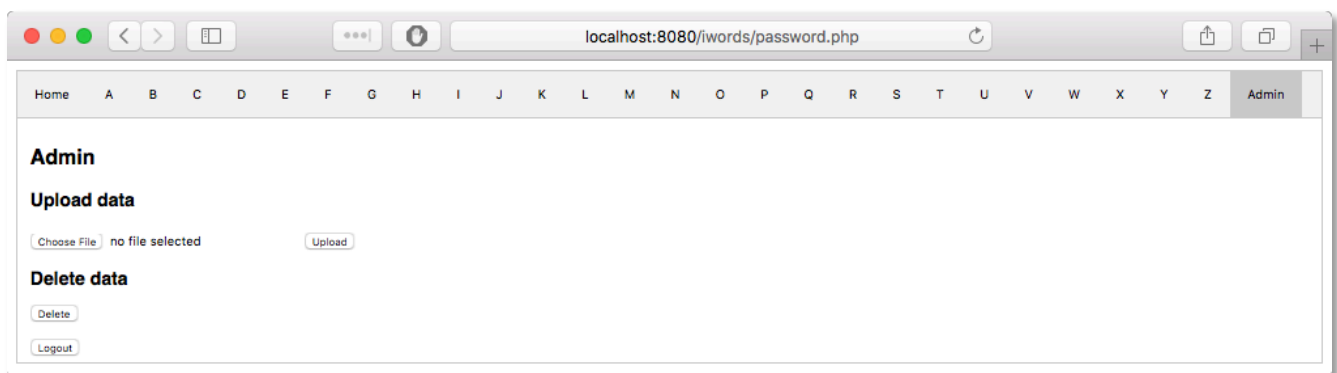


Figure 6. Web page showing the administrator's options.

To upload a file, administrators indicate a local file by using the Choose File button (or similar in other browsers) to locate it. Once a file is chosen, pressing the Upload button displays a JavaScript dialog (below, in this case for file "abc.txt") to confirm the request. Clicking OK uploads the file. Clicking Cancel goes back to the admin options.

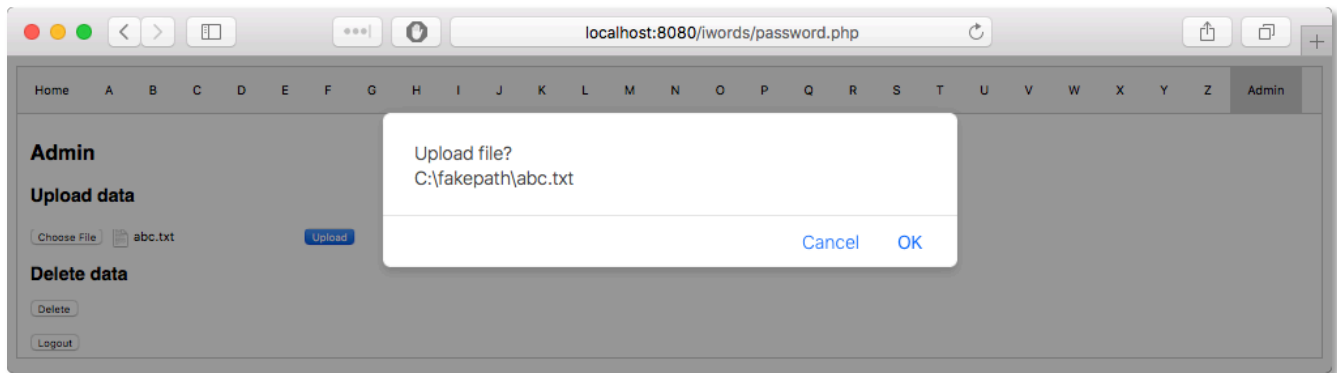


Figure 7. JavaScript dialog asking confirmation before uploading a file.

Pressing the Upload button without choosing a file displays a JavaScript dialog (below) with an error message.

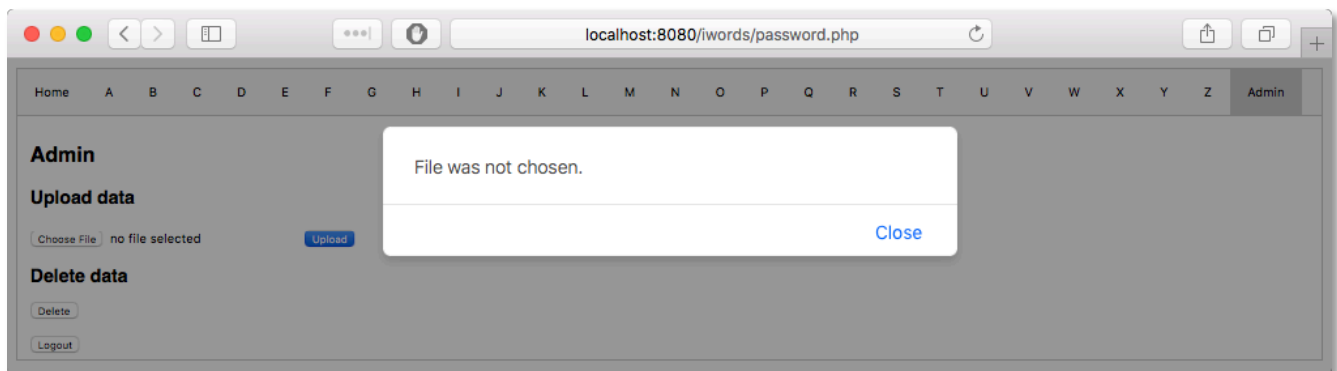


Figure 8. JavaScript dialog indicating that the Upload button was pressed but a file has not been chosen.

Note that the visual components to select and upload a file are different across web browsers, and thus your implementation may look different.

After a file is uploaded, a web page (shown below) will indicate the number of entries added (50,495 in this case). Pressing the Back to Admin button (at the bottom of the page) redirects back to the admin options.

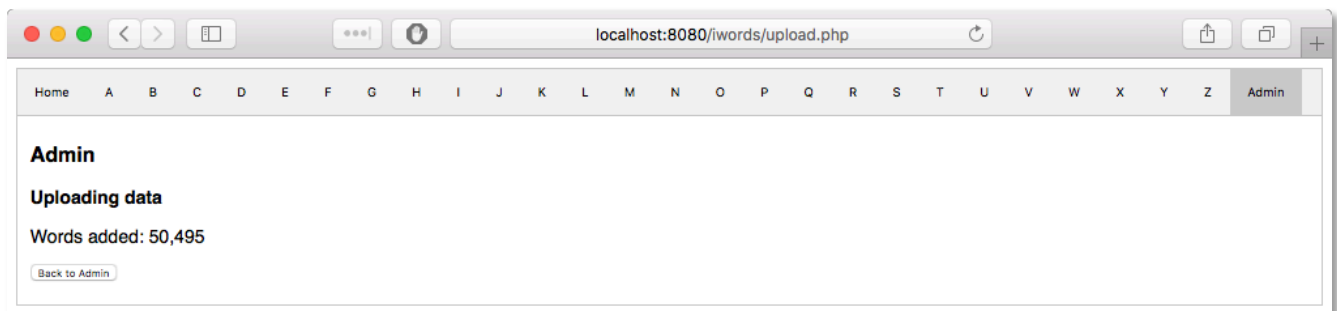


Figure 9. Web page showing the number of entries added from an uploaded file.

Choosing the Delete button (under Delete Data in the Admin page shown after login) deletes all words from the web site. This option shows a JavaScript dialog (below) asking to confirm the request. Pressing Cancel leads back to the Admin options without deleting any entries.

Developer Note: Any transition between the Admin options and the upload and delete pages should keep track of the administrator's password (which is needed by MySQL to access the database). You can use a hidden field in forms or use sessions. If using sessions, your code must keep track of different concurrently logged in administrators by generating unique keys stored in session maps. Ask if unsure about this feature.

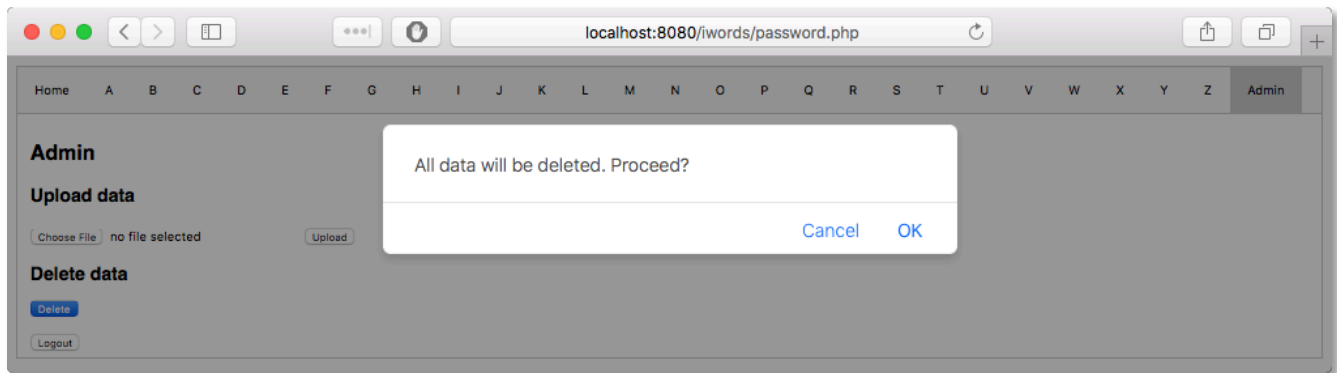


Figure 10. JavaScript dialog asking confirmation before uploading a file.

Pressing OK displays the page below indicating all words were deleted. The Back to Admin button leads back to the admin menu.

Developer Note: Deletion entails the removal of words from tables, not the removal of tables from the database (which in any event won't happen on my computer because database drop table privileges will be disabled).

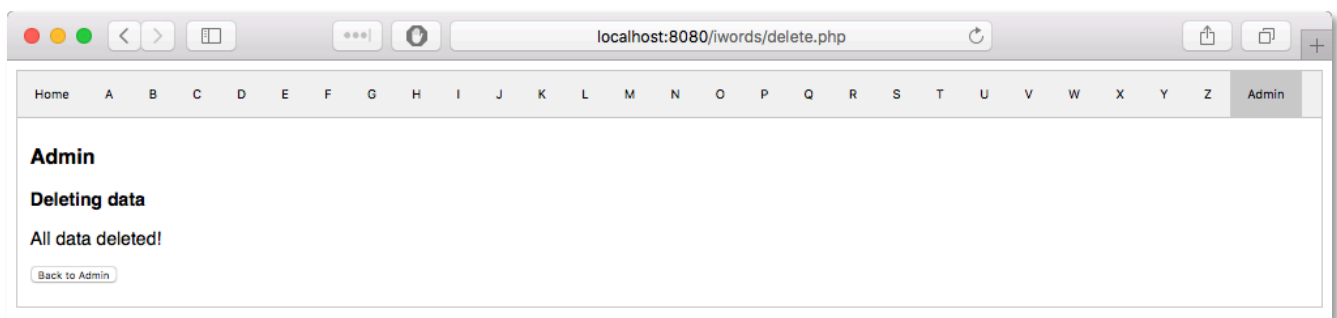


Figure 11. JavaScript dialog asking confirmation before deleting all words.

As shown below, no words are listed once all data has been deleted.

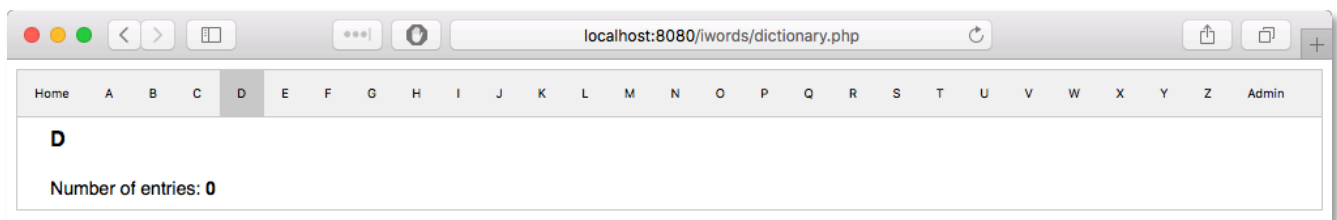


Figure 12. Web page showing no items exist that begin with letter D.

Data

Use MySQL to store all words displayed by the website.

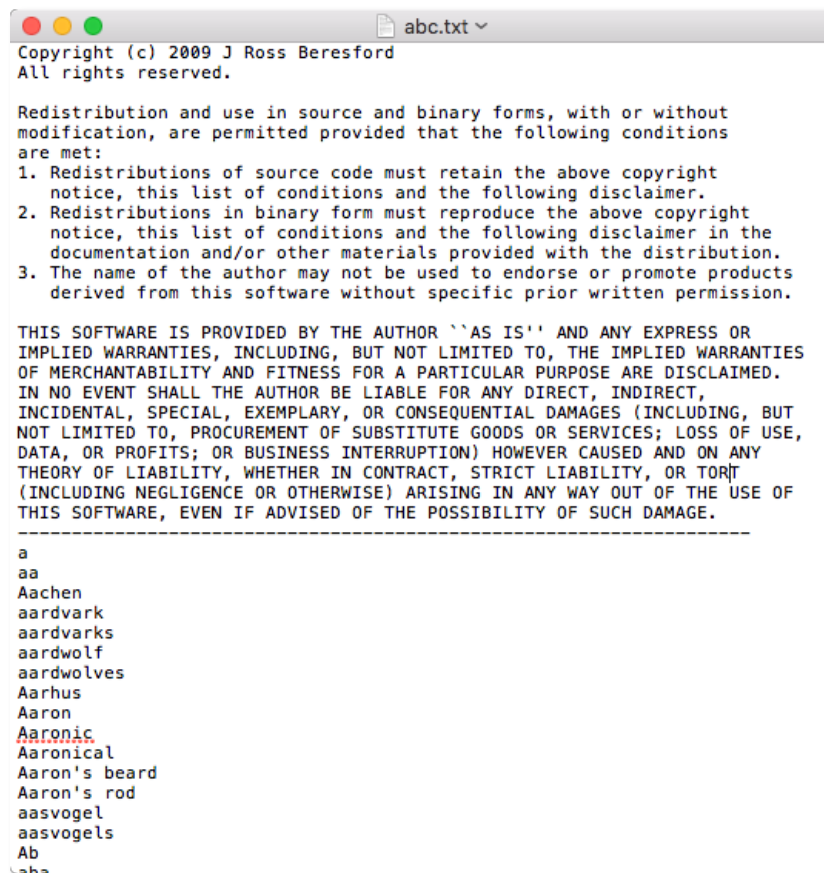
To this end, create a database (named **words**) with one table (named **wordsTable**). Use the MyPHPAdmin tool shown in lectures to create it. The table defines one field (named **entry**) of type text with a length 100. Make it be the primary key (to prevent duplicates).

In addition, create user accounts named **wordsroot** and **worduser**:

- *worduser* is the username to access words (e.g., when selecting letter tabs). It has localhost access privileges to read table information only (SELECT). Its password is **anonymous**.
- *wordsroot* is the administrator username for uploading and deleting data. It has localhost privileges to write table information (INSERT, DELETE). You choose this password, which (as mentioned in an earlier Developer

Note) should not be hardcoded anywhere in your code; rather your code should receive it in the login page and pass it along the upload and delete scripts (and vice versa).

Text files (as shown in the figure below) start with a copyright notice that ends with a line made of dashes. After this marker, there is a list of text lines, where each line becomes one entry in the database. The copyright notice can vary from file to file but all end with dashes.



```
Copyright (c) 2009 J Ross Beresford
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products
derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR
IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,
INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
-----
a
aa
Aachen
aardvark
aardvarks
aardwolf
aardwolves
Aarhus
Aaron
Aaronic
Aaronical
Aaron's beard
Aaron's rod
aasvogel
aasvogels
Ab
aba
```

Figure 13. Text file used as data source to upload database entries.

Note that lines with words may contain blank spaces (if the entry is made out of several words) and special characters (such as quotes and apostrophes). For example, the line “Aaron’s beard” is a tab A entry (it begins with “A”) with a blank and an apostrophe.

Initial Files & Additional Info

The assignment comes with the following initial files (downloaded as a single ZIP file “words.zip”):

- CSS style sheets (“iwords.css”)
- HTML pages (“index.html”)
- data files (several text files with words).

SQL

To read all entries that start with a letter (for example ‘A’) you can use the SQL statement:

```
select * from wordsTable where entry like 'A%'
```

HTML

To create a table where rows get highlighted when hovering over with the mouse (using the provided css file):

```
<table class='words'>
<tr class='highlight'><td>This row</td><td>gets highlighted</td></tr>
</table>
```

Groups

This assignment is done in groups of two students assigned by the instructor.

No code (partially or totally) should be taken or disclosed to/from other people (except your partner, if you have one), including online resources. If in doubt, read the “Honor Code” section in the syllabus and contact your instructor.

Submission

Assignments will be presented in person at the designated times.

Bring your code to your appointment time. You should explain all decisions taken to arrive to your solution.

A successful assignment is one that implements the functionality outlined in this assignment description.

...