|  |
| --- |
| **PRACTICAL ASSESSMENT** |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Student Name:** | | | **Angelo Palazzolo** | | | **Student Number:** | | | **3150704699** | |
| **Unit Code/s & Name/s:** | | | ICTDBS504: Integrate database with website | | | | | | | |
| **Assessment Description:** | | | Practical Assessment | | | | | | | |
| **Assessment Due Date:** | | |  | | **Assessment Received Date:** | | | | |  |
| **Student Declaration:** I declare that this assessment is my own work. Any ideas and comments made by other people have been acknowledged as references. I understand that if this statement is found to be false, it will be regarded as misconduct and will be subject to disciplinary action as outlined in the TAFE Queensland Student Rules. I understand that emailing or submitting this assessment electronically confirms my assent to this declaration in lieu of a written signature. | | | | | | | | | | |
| **Student Signature:** | |  | | | | | **Date: / /** | | | |
| **Assessor Feedback:**  ☐ **Student provided with feedback** | | | | | | | | | | |
| **Attempt 1 Satisfactory ☐ Not Satisfactory ☐ Date:      /     /** | | | | | | | | | | |
| **Attempt 2 Satisfactory ☐ Not Satisfactory ☐ Date:      /     /** | | | | | | | | | | |
| **Assessor Name:** | D. Batchelor | | | **Assessor Signature:** | | | |  | | |
| **Note to assessor: Please record any adjustment below that has occurred to this assessment e.g. Oral questions were used to support student’s ability to complete tasks identified.** | | | | | | | | | | |
|  | | | | | | | | | | |
| **Disclaimer:** TAFE Queensland North (TQN) is collecting the information on this form for assessment purposes. Only authorised Departmental officers have access to this information. Your personal information will not be disclosed to any other third party without your consent, unless authorised or required by law, in accordance with the Information Privacy Act 2009. | | | | | | | | | | |

**This section must be completed by the student when the student either posts or hands a completed assessment/assignment to customer service centres. a receipt is only issued for face to face assessments when requested by the student.**

• **RECEIPT**

|  |  |  |
| --- | --- | --- |
| STUDENT TO COMPLETE |  | OFFICE USE ONLY |
| Unit of competency code/s:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
| Assessment item/No: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | Date received: / / |
| Student’s name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |

|  |  |
| --- | --- |
| **Instructions to Student:** | **Time allowed:** This assessment is to be completed over the semester.  **Materials and equipment to be supplied by the Student:**   * Access to a PC, Web Authoring Tools, web server and hosting.   **General Instructions:**   * Review technical specification of project * You have 3 tasks to complete. * You will need to answer **all** questions correctly * Keep a copy of all assessments as you will be required, to submit them as a GITHUB repo to your teacher.   NOTE: If you are not successful on your first attempt, you will be given comments and a further two weeks to rectify the changes.  You will need to re-submit your corrections for remarking. If you are unsuccessful on your second attempt, you will be required to re-enrol in this unit. |
| **Instructions to Assessor:** | Please see Instructions to Student above. |
| **Assessment Context and Description: (if required)** | In order to gain competency in ICTDBS504: Integrate database with website, successful completion is required of the following assessments items:   * Practical Assessment |
| **Note to Student:** | An Assessment Matrix is available from your teacher upon request. The matrix shows how the knowledge and skills that you are being asked to demonstrate align to the Elements, Performance Criteria, Critical Aspects, Required Skills and Required Knowledge components of each Unit of Competency. |

**Project**

You should create a new repo called '**assessment1\_sem2\_yourname** in GITHUB account and save all files in this folder. You should create sub folders as necessary.

**Technical Specifications**

You are required to create an address book web application that stores contact details (First name, last name, phone numbers, email address, Facebook profile) of a user’s contacts.

The web application should have:

* A secure login
* The ability to add contacts.
* The facility to look up contacts by last name
* The ability to update and delete contacts
* Data validation that checks user's input;
* A user friendly GUI.

## Task 1

***You are required to:***

* ***Identify the data needs of the web application, and outline the following information:***
* ***What data needs to be submitted to the application’s database through the GUI?***

The application is designed to used by a single person, and as such only a single password exists to enter applicication. Hence the user must enter the correct password to gain access to the data and features of the application. (See figure 1 below.)

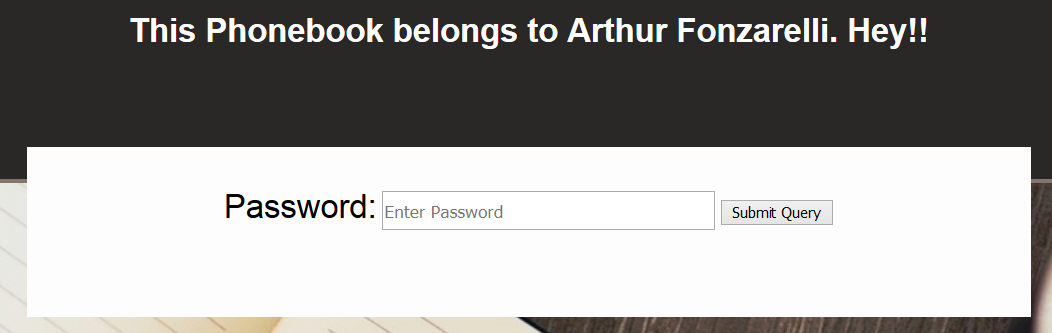


Figure 1: User must enter password to proceed

The application provides the user the facility to create and add new members (contacts) to the database of contacts. Figure 2 (below), displays all the data collected for each member. The user enters the appropriate information in the relevant field. Three items of information (identifed with a “\*”) are set as “required” and as such the application will not continue until correctly formatted data is entered into these fields.

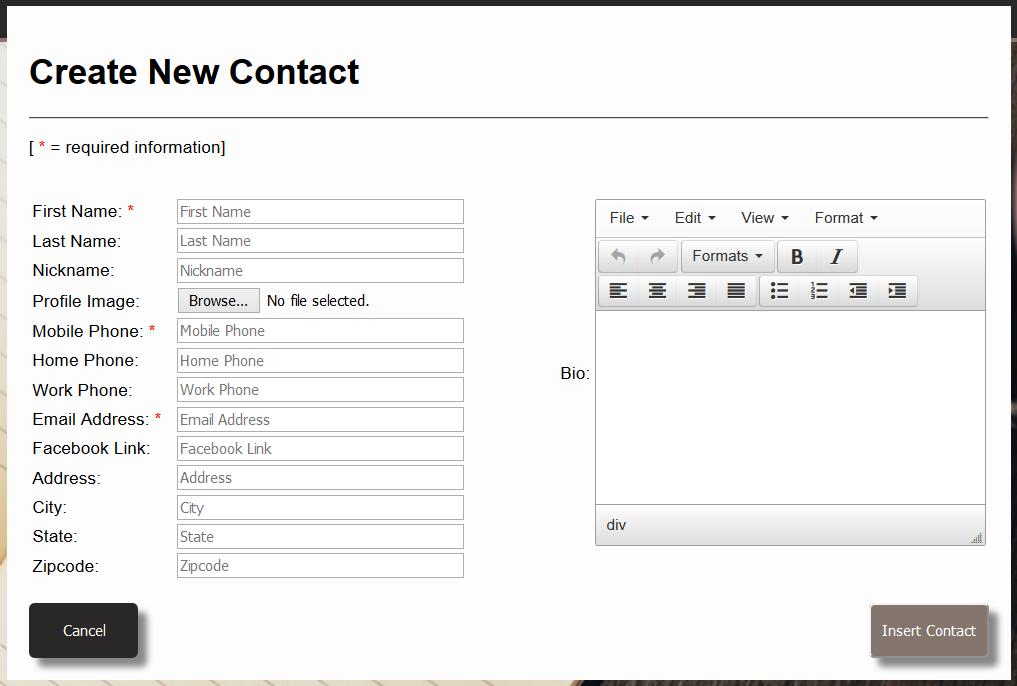


Figure 2: The 'Create New Contact form, displaying data to be entered by user.

* ***What data needs to be retrieved by the application for display in the GUI?***

The application displays in two different screens. The main GUI screen seen below in figure 3, presents a list of all current contacts within the database, but only displays three pieces of information – name, nickname and mobile phone number – for each. When this screen is displayed the php code accesses the database and retrieves the three pieces of information for each contact that is listed as current (ie. a contact that has not been deleted/archived).

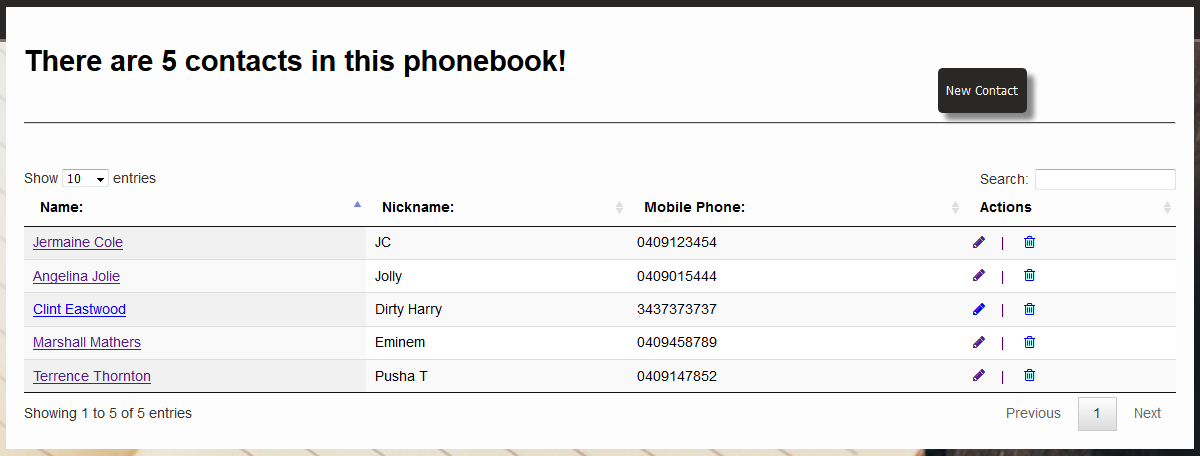


Figure 3: The main (homepage) GUI displays three pieces of information for each contact.

The second GUI screen, seen below in figure 4, presents all the information stored on the database for each contact. To see this screen the user clicks on the individual contact’s name on the main screen. The php code accesses the database and retrieves all the data for that contact based on their “contact\_id” (ie. the unique number allocated to each new member added to the database).

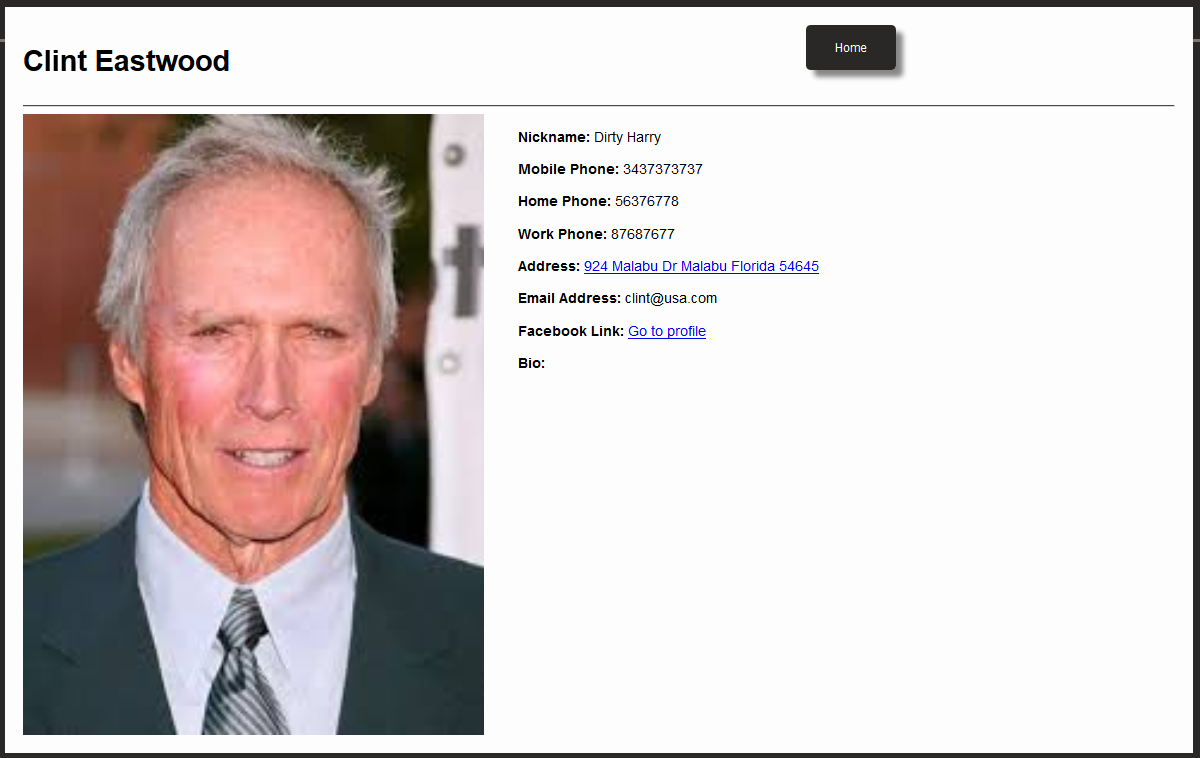


Figure 4: The complete profile containing all information (including photograph), for each contact, can be displayed.

* ***Explain how the following programming concepts will be utilised in this web application:***
* ***Authentication and security.***

Authentication is achieved via the user login (seen below in figures 5 and 6). As explained earlier, the application is accessible by only one user with on specific password. This password must be entered when the application is launched. The php code checks the password entered by the user, and compares it to the one stored in the php code (which resides on the server)

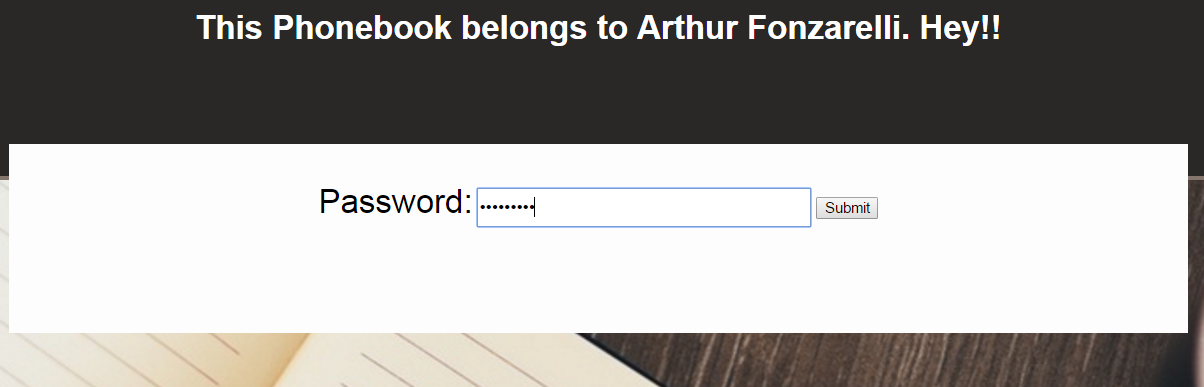
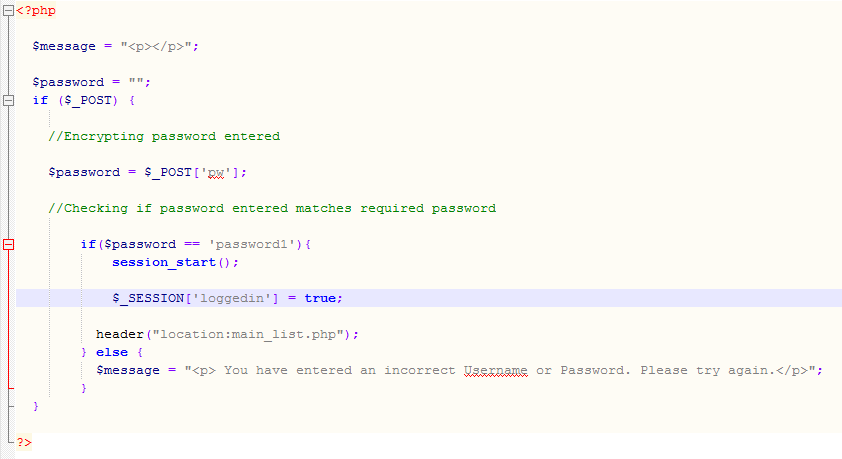


Figure 5: “login.php” – user must enter password to proceed



Each page that requires the user to enter data, first checks for the existence of the session variable created at login. If not, user is redirected to the login screen.

Checks is a password has been entered

User is directed to the main screen

If it matches a ‘session variable is created to be used on subsequent screens to verify user has logged in

Checks if the password entered matches the required password

Figure 6: "login.php" - session variable is created.

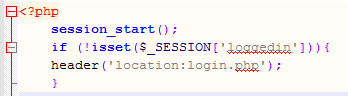


Figure 7: The “absence” of the session variable (login) is checked before proceeding.

To prevent the insertion of malicious code, all fields requiring users to input data, php code first modifies the data entered, thus removing specific characters required enable the malicious code.

This php code is also used to ensure that is entered (ie the field is not left empty) and that the data entered, addresses specific requirements and formatting. This is visible in the two figures below.

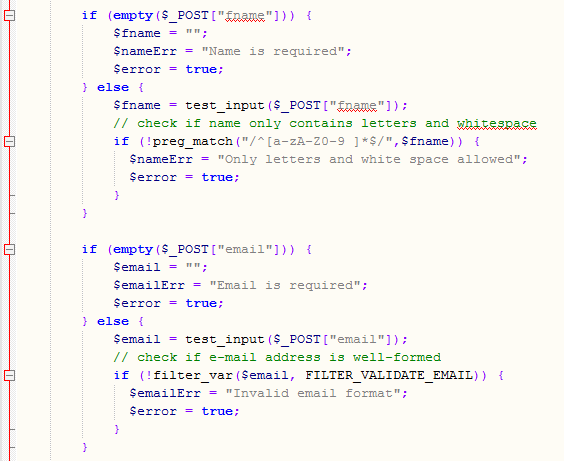


Figure 8: The data entered is validated using php scripiong.

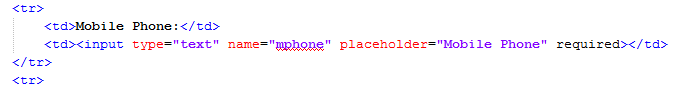


Figure 9: Data is validated (marked compulsory) by html code.

* ***HTTP.***

HTTP is the underlying protocol used by the World Wide Web. HTTP defines how messages are formatted and transmitted, and what actions Web servers and browsers should take in response to various commands. For example, when you enter a URL in your browser, this actually sends an HTTP command to the Web server directing it to fetch and transmit the requested Web page.

The application function through a web browser interface, and as such, in many respects functions as would a webpage. Thus HTTP is a crucial means to achieving functionality.

* ***Session management.***

A web session is a sequence of HTTP requests and response. Modern and complex web applications require the retaining of information for the duration of multiple requests. Php coding is used to establish variables that retain this information which will apply to each and every interaction a user has with the web application for the duration of the session.

Figures 6 and 7 on page 5 outline this web application makes use of sessions once the user has authenticated. This ensures the ability to identify the user on any subsequent requests (or screen).

* ***stateless/stateful programming.***

A computer (program) must store information/data in variables, which represent storage locations in the computer's memory. The contents of these memory locations is called the program's state. A computer systems possess the two states of ‘stateless’ and ‘statefull’.

With the stateless state, the data entered by the user is held by the computer or program, for as long as it is needed for the current task or until the connection to the computer or program is broken. After this point the information entered is removed from the system.

With the statefull state, the data entered by the user is held by the computer or program indefinitely, or until the computer or program is terminated.

The application is predominantly stateless. The appearance is controlled by HTML and Javascript which is loaded every time a page link is activated. The database content is again stateless with the php code accessing the database to propagate the tables, every time required screen is activated. The only stateful component is ‘session variable’ (see figures 6 and 7 on page 5) created at login. This session variable remains “alive” as long as the application remain running, and is referred to when a majority of screens are opened to check/ensure that the user has in fact logged into the system.

## Task 3 - Write a short brief on your web application

***You are required to write a short report on your web application of no more than 5 pages (including screen shots). The report is to cover:***

* ***The web application's functionality.***

1. The user enters the appropriate password to gain access (See figure 5, page 5)
2. Php code checks that the password entered matches the required password.
3. A ‘session variable’ is created to verify that the user has logged into the system (See figures 6 & 7, page 5)
4. The main screen lists all the “current” contacts; by displaying their name, nickname and phone number.(See figure 3, page 4)
5. The user can access additional information about an individual contact by clicking on his/her name. This brings the user to another screen which displays all the information about that contact, as stored on the database, including a photograph (if available). (See figure 4, page 4).
6. Clicking the ‘Home’ button returns the user back to the main screen.
7. From the main screen the user can create a new contact by clicking the ‘New Contact’ button. This brings the user to another screen (see figure 2, page 3) containing a blank form. The user enters relevant data. Three of the fields are mandatory (name, mobile and email). The fields of ‘name’ and ‘email’ are and made secure by php code that controls the content and/or format of the data entered by the user. (See figures 8 & 9, page 6). Clicking ‘Insert Contact’ saves the data entered into the database.
8. From the main screen the user can modify or update the data pertaining to an individual contact by clicking on the pencil-icon. This brings the user to another screen (see figure 10, below) displaying a form that contains the existing data. The user delete/type over the data they which to save relevant data. Clicking ‘Update Contact’ will overwrite the data in the database with the content in the update form.

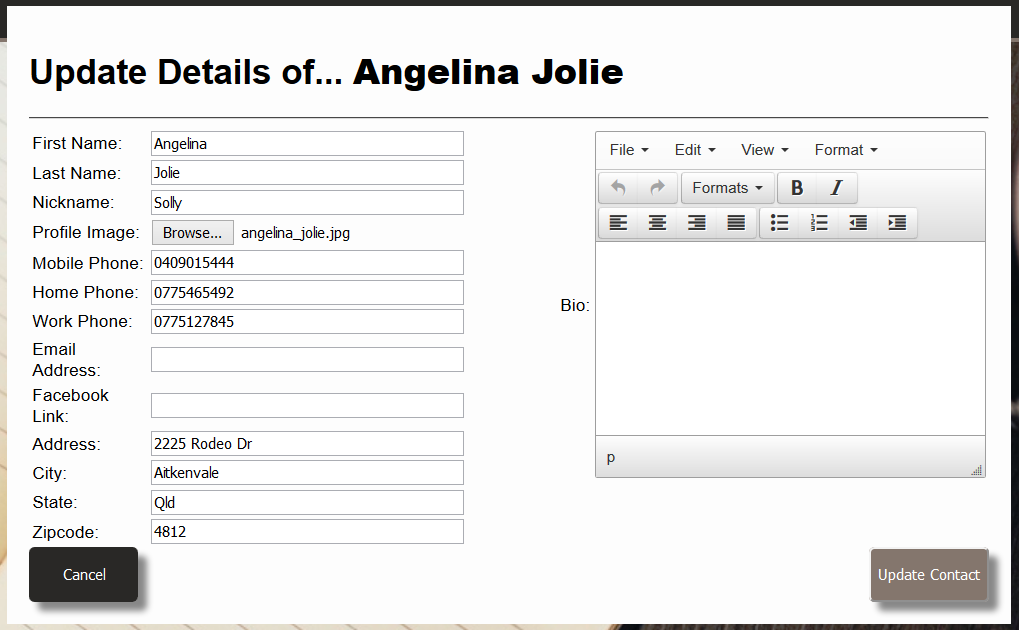
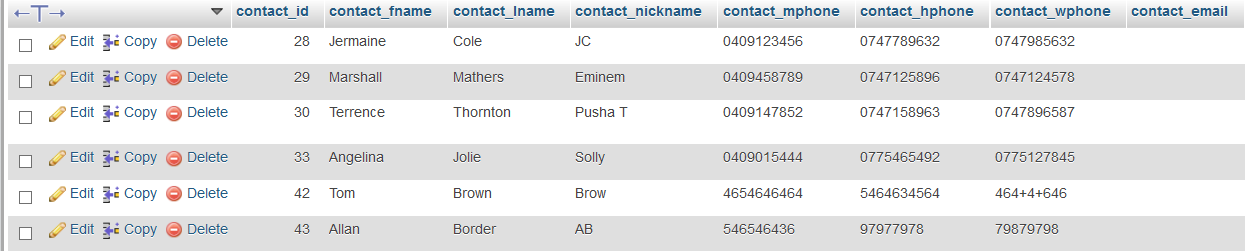


Figure 10: 'Update' screen displaying existing information

1. From the main screen the user can “delete” an individual contact by clicking on the rubbish bin. While this function is called “delete”, the actual data remain in the database, but has its status changed to ‘archived’ and will not be displayed on the main screen. At this stage, reinstating the contact as ‘current’ will have to be done through the back-end, using SQL code.



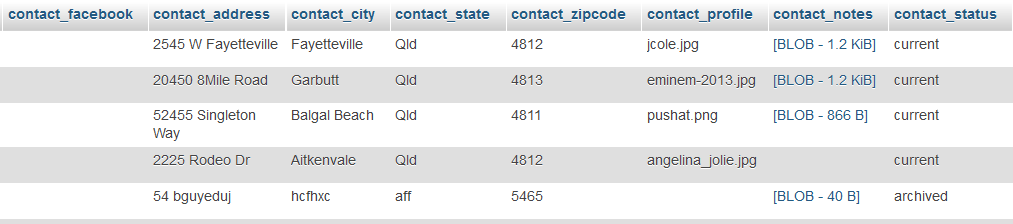


Figure 11: Contact's status had been changed to 'archived' and thus they have been "deleted" even though their data remains in the database.

* ***The database structure; and***

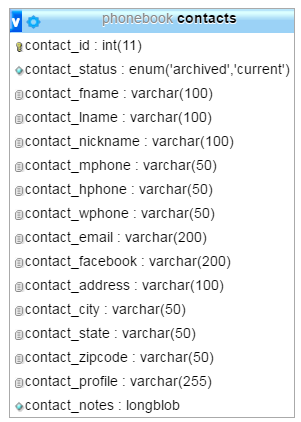


Figure 12: The fields/attributes of the "contacts" table, that constitutes the "phonebook" database.

* ***The php/SQL code you used to interact with the database;***

index.php

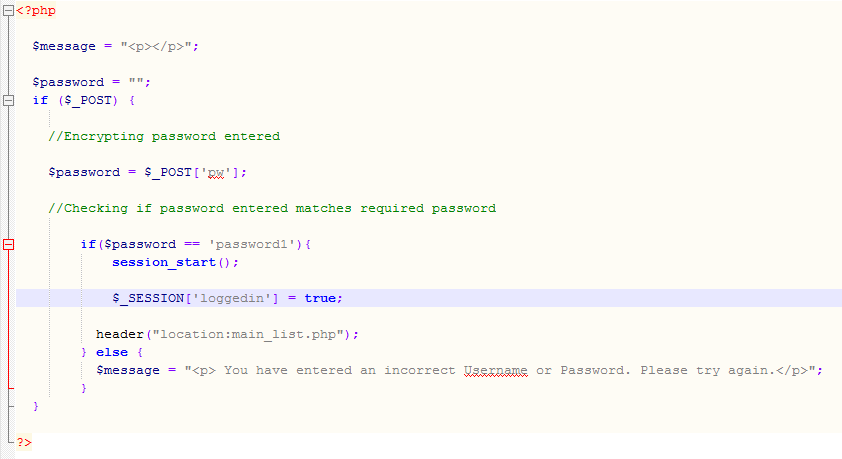
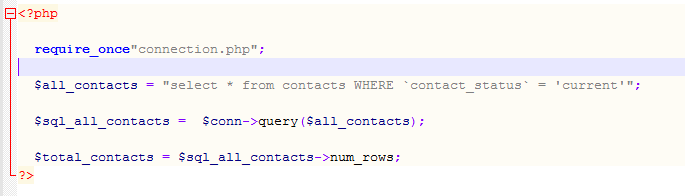


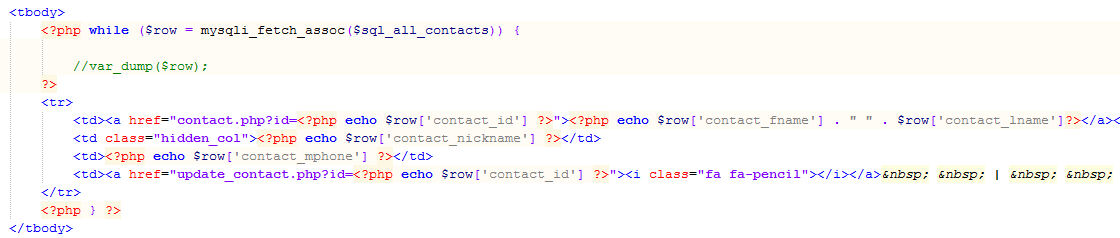
Figure 13: "index.php" - PHP code ensures authentication. See figure 6 & 7, page 5, for explanations



A connection is made to the database.

SQL code selects only the contact whose status is "current" to display in the main screen.

Figure 14: "index.php" - SQL code selects only the contact whose status is "current".

main\_list.php

php sets an array from the data collected from the query at the top of the page. The data in the array is then used to populate the table for the main screen.

An alert message verifies the successful “deletion”. On clicking ‘ok’, the user is directed to the main screen

SQL code used to update the status of the contact that is being “deleted”

Php checks if the password entered matches the required password

SQL code selects only the contact whose status is "current" to display in the main screen.

The application first checks for the existence of the session variable created at login. If not, user is redirected to the login screen.

Figure 15: 'main\_page' - SQL and php code

insert\_contact.php



Data enter in the form by the user is prepared for the database

SQL determines if a specific image exists within a specified folder. If not, a generic imager is allocated.

SQL creates a new contact and inserts the new data into the database

This php code sets the field as ‘required”. It then checks that the address entered is formatted correctly.

This php code sets the field as ‘required”. It then sends the field to a function that removes specific characters to ensure it does not contain malicious code. I then checks that it only contains letters, numbers and spaces

The application first checks for the existence of the session variable created at login. If not, user is redirected to the login screen.

Figure 16: "insert\_contact" - SQL and php code to create a new contact and add to the database

contact.php

The application first checks for the existence of the session variable created at login. If not, user is redirected to the login screen.



The data collected in the array, is placed into a table.

A connection is made to the database.

SQL gathers all data belonging to contact with a specific “ID”.

The data collected is placed into an array.

Figure 17: "contact.php - php and sql code used to display the data of an individual contact.

update\_contact.php



SQL creates a new contact and inserts the new data into the database.

SQL overwrites the existing data in the table, with the data contained in the update form.

SQL determines if a specific image exists within a specified folder. If not, a generic imager is allocated.

All the data in the form (existing and newly entered by the user) is prepared for the database

A connection is made to the database.

SQL gathers all data belonging to contact with a specific “ID”.

The data collected is placed into an array.

The application first checks for the existence of the session variable created at login. If not, user is redirected to the login screen.

The report is to include screenshots of the GUI and screen shots of the above points.

## You will need to submit

The following should be included in your GITHUB repo (please ensure that there are no additional files):

* Answers to task 1
* Completed web application
* Web application report