Advanced Higher Computing Science Project

Project Proposal

A group of students would like to be able to communicate with their friends and classmates outside of school about their school work and other school matters. The website will allow students to register their information such as email, name, class, date of birth, and gender, along with a username and password to identify their account. Once registered, students will be able to access their own profile and change their details should any need to be. They will also be able to access a main message board where they will be able to post messages addressing whatever their concern is. Should any message be particularly offensive, they will be able to submit a complaint on a report form for review from a system admin.

The user group will be students of mixed ability, gender, and age. For this reason the website will be relatively straightforward to navigate for people of all levels of computing skill.

Advanced Higher criteria that this project will satisfy are as follows:

HTML form processing using server-side scripting
In order to register for the site, students will have to enter their details to a registration form. The data submitted to the server through the form will have to be processed and stored into the appropriate field in the database. This will require the form to be processed using both PHP operations and communication between PHP and HTML.

Appropriate SQL operations
The server-side script will make use of SQL queries to handle the data entered in the registration form in order to place it into the database. It will also use a query for updating users details if and when they choose to use this facility. Logging in will also require a SQL query to check that the data entered matches a record in the database. The deletion form will require a ‘drop’ SQL query in order to delete the record from the database.

Ben Manson

Analysis

As part of my analysis of the task at hand I will carry out user surveys of members of my target audience - pupils in my school. The completed user surveys are included. The other aspects of the analysis included are a short feasibility study, and my analysis of the user survey findings and how they will be used to help develop the project.

Feasibility Study

Legal Feasibility

To make sure that the project doesn't break any laws under the Copyright, Design and Patents Act, it will be made sure that any logos used are created by me or someone I know who is willing to allow me the use of them.

As a database will be used to store people's information the project must keep within the laws detailed by the Data Protection Act meaning that users can at any point request to see the data that is stored about them and can also make changes to it upon request.

A complaint form will have to be available in order to monitor the content posted by users of the website. This means the project can keep within the Communications Act to allow all users to feel safe and assured when using the site.

The project will be stored on and run from an offline pen drive. This will make it incredibly difficult – but not impossible - to be breached. However, this risk cannot be considered great enough to deem the project unfeasible.

As the project will most likely never be made available to the public, it is highly unexpected that the authorities will ever contact me demanding that I present the data stored as most of it will be made-up details. However, should they request such information it would be available for their analysis.

Time Feasibility

The project deadline is on the 21st March 2017. The main milestones for the development are as follows:

- **Requirements Specification – 4th November 2016**
- **Test Plan – 18th November 2016**
- **Interface Design – 25th November 2016**
- **Program/Data Structure Design – 2nd December 2016**
- **Implementation – 28th February 2017**
- **Testing – 10th March 2017**
- **Evaluation – 17th March 2017**
- **Final Submission of Project – 21st March 2017**

If all milestones are hit as predicted then the project is feasible in terms of time.

Economic Feasibility

There is no money available to spend on the development so any software used for the implementation will have to be freeware. It should also be open-source to prevent any

need for licensing or payments that could be demanded by the developers of the software in order to release the end product.

Technical Feasibility

In terms of the hardware needed for the project, it will be run from a pen drive so this is all that will be required. Most of the technical aspect of this particular project is in terms of software and skills needed to complete the implementation. My skills in the particular languages that will be used to develop the website – HTML, CSS, PHP, SQL – will have to be improved, but I can use various websites resources to help me on the tougher sections of code.

User Survey Analysis

Due to the survey being carried out on school time, the fifth and sixth year students were the most accessible to be surveyed. However, I was able to give the survey to at least one pupil from all other year groups. All of the students that filled out the survey said that they has used social media, and a majority had used message boards to aid their studies. This made sure that the product would be something that students would find useful.

Due to the split in people finding social media's colourful interactive scheme more engaging and people finding a plain message board format more helpful, I have decided to try and combine the 2 formats and design a message board with an engaging colour scheme.

In Q4 the students were posed the question of whether they feel comfortable with large social media companies storing their data. This question was again met with a mixed response with some saying they do feel comfortable and some saying that they don't. To me, the appropriate response to this is to include a brief message on the homepage stating to pupils that the only data that will be stored about them is the information that they entered when they registered with the site.

An element of the website that had not yet crossed my mind was the safety of the users. The overwhelming response by the students that safety was very important to them led me to think about features that could help with student safety. The simplest and perhaps most effective way to counter this is to add a complaint form somewhere in the project.

NOTE: A blank user survey along with all completed user surveys are include.

Project Plan

Outline Project Plan

TASK	TIME (HOURS)	TARGET DATE	RESOURCES
Requirements specification	5	04/11/2016	End users, analysis
Test plan	3	18/11/2016	Requirements spec
Interface design	6	25/11/2016	Wireframes
Program/data structure design	5	02/12/2016	Pseudocode, structure diagram, data dictionary, user case
Implementation	50	28/02/2017	HTML editor, database, server side editor
Testing	5	10/03/2017	Test plan, final product
Evaluation	5	17/03/2017	Requirements spec, end product
Final submission of project	2	21/03/2017	Evidence of all of the above

Detailed Project Plan

Requirements Specification

For the requirements specification I will first identify all user input that the site should be able to receive along with the validation that will be required for it. Each page that will be included in the final solution will be listed along with the input from that page.

I will then identify the processes that will be carried out by the server to handle this input data. These will be listed in a process section below the inputs.

I will finally identify the outputs that the site will produce. This will include errors displayed by validation, and any output produced using the user's personal data i.e. the profile page. The pages that produce output will be listed and in each section any site/page output will be listed.

Test Plan

To start the test plan I will identify the inputs that will have to be tested using the requirements spec. As it is to be comprehensive testing, every single input in the requirements spec will be listed. For every single input that is to be tested I will then identify suitable data that can and will be used to test it. This will most likely involve various stages for each input.

Interface Design

When designing the interface I will first design a general layout for all pages using the requirements spec and user survey analysis. Using this layout I will be able to draw up a wireframe for all of the pages accessible to the user. I will then add validation to these

wireframes by annotating each individual input with how it will be validated. I will complete the interface design by adding a short description below every page stating exactly what the main purpose of the page will be.

Program/Data Structure Design

The first task that I shall carry out during program/data structure design will be to complete a structure diagram for the site. This will be a sheet of A4 paper with all pages that will be required for the site to be functional positioned so that the links between them can be indicated using arrows.

I will then design a data dictionary for the tables that will be required in the database for the site to function correctly. This will indicate data types, length, whether it's required, and constraints for all listed fields along with auto increment and primary/foreign key for the appropriate fields.

Then, I will design pseudocode for the server-side code and the client side validation. This will make it much easier to implement during the build of the site as I will have a general blueprint to follow for most of the difficult sections.

To conclude the data structure design I will design a use-case diagram to show how various different actors will interact with the site. This will include users (registered and unregistered) and the different tables in the database.

Implementation

To begin construction on the site I will design the HTML pages. This will basically just be the forms that require user input as most other pages will require PHP server-side code. This will require me to add some HTML defined commands to validate some of the inputs.

I will then create the PHP files to handle the server-side scripting. This will include pages where the data inputted to forms is inserted into the database as well as some session config files. The next step from this is to add the SQL queries into the PHP files. These two will run alongside each other add I will add the SQL queries as I create the PHP pages.

The second to last step in the implementation will be to add client-side validation to the user data forms. This will be implemented using JavaScript files. Once the JavaScript has been coded I will return to the individual HTML form pages and link them to the JavaScript.

The last step in the implementation will be to style the site. This will be implemented using CSS to keep a consistent layout across all pages. The wireframes design will be used to keep this layout consistent. I will then return to all pages - PHP and HTML - and link the stylesheet to them to style them.

Throughout the implementation I will test all the elements that I build as I build them in order to make sure that it works properly. The whole site will be comprehensively tested during testing after the final product has been built.

Testing

Before beginning the actual testing of the site I will design tables containing the tests and data stated in the test plan along with the expected result and whether the actual result matched it.

Before carrying out my own comprehensive testing I will design a small table of tasks for my end users to carry out. I will then gather a small group of students to carry out this user testing. The results will be analysed in the evaluation at the end.

To conclude the testing I will carry out the tests stated in the test plan and record the results in the tables created at the beginning of testing. This will include making sure that the expected output matched the actual output.

Evaluation

The evaluation will involve 3 main tasks: evaluating the testing; evaluating my performance, and evaluating the completeness of the solution. When evaluating the testing, the main focus will be whether the results matched the expected results. I will also write up a quick evaluation of the user testing.

When evaluating my performance I will annotate a second copy of my project plan (Gantt Chart). This will help identify whether or not I met milestones as predicted, and if not why not and how it affected the final development.

When evaluating the completeness of the solution I will mainly focus on whether the solution met all requirements in the requirements specification. I will also write a short bit about any improvements that could be made.

When all above tasks are completed as the stated the project will be completed and evidence will be gathered for submission.

Requirements Specification

The project will feature many pages, all of which can be accessed once the user has logged into a registered account. The registration page and the login page will be the only pages able to be accessed before a user has logged in. Once they have logged in and the session has been initialised, the other main pages will be able to accessed. These will be: the homepage featuring information about the project; the user specific profile page featuring the information that they registered about them self; and the message board page featuring messages posted by all users. Users will be able to submit a complaint about a message should they feel unsafe or offended by it. The data inputted by the user will be processed using a combination of HTML, PHP, CSS, JavaScript, and SQL and certain aspects of it shall be outputted across the pages of the site.

Inputs

Homepage

The homepage will be a plain page with links to the message board and the profile page. It will display some information about the website and also a welcome message to the user who has logged in. It will require no input from the user.

Login

Upon accessing the website, the user will be taken to the login page automatically. From here they are presented with a choice of accessing the registration page or logging in. They will not be able to access any other page on the site until they have logged in. To login they will have to submit their login details to a form. These details along with their validation will be:

- Username - must be existing in user table
- Password - must match the record containing the username entered

It should be noted that none of the fields in the login form will be able to be submitted empty.

Registration

Should the user choose the registration page they will be presented with a form to fill in with their details that will be submitted to the database should it be appropriate data. Should they choose the registration page by accident then they will be able to return to the login page, but - like with the login page - they will not be able to access any other pages from this point. The form inputs along with their validation will be:

- First name - maximum length of 15
- Surname - maximum length of 20
- Gender - radio button, choice of 2; male and female
- Date of Birth - date choice from calendar; must be >1997 and <2006
- Email address - will be validated for appropriate email syntax (@ and .co)
- Username - maximum length of 15
- Password - must be between 8-20 characters and contain a number, lowercase character and uppercase character

It should be noted that none of the fields in the registration form will be able to be submitted empty.

Complaint

The complaint form will be a simple complaint form where the user will enter their complaint into a text box. The user will be instructed what details to include in the complaint, along with their actual complaint about the message. The contents of the text box will then be stored in a database and reviewed by a site admin. The only validation on this page will be a maximum length of 140 characters on the complaint form. This will eliminate any storage issues that could arrive should a user submit multiple huge complaints.

Message Board

The actual message board itself will require two pages: a page for the user to enter and post their message, and a page to display all users messages. The messages will be displayed in a table showing details such as the content of the message, the date and/or time it was posted and the username of the user who posted it. When posting the message the user will simply enter it to a text box. Like with the complaint form, the only validation required will be a maximum length of 140 characters on the message. This is again to eliminate storage issues that may arise. On the message board there will be links to the post page and the complaint form page.

Profile

The profile page will be a fairly basic profile page showing the logged in user the details about them in the database that they are able to update. This will be their name, gender and date of birth. Students will also be able to update their password, but for obvious reasons this will not be displayed in their profile. On the profile page there will be links to the update page for each of these details.

Update

There will be an update for each of the details stated in the profile section that can be updated. Each of these pages will be constructed exactly the same way. They will all contain a form asking them for the specific detail to be edited and can then submit this. The validation on whatever detail is to be edited will be the same as the validation stated in the registration section.

Processes

The server will carry out various different processes to allow the user to access the website and then navigate it as intended. When a user registers for the website, the server will transfer and store the data they entered in the form to the appropriate table using a combination of PHP and SQL queries. When a user logs in the server will setup session variables to allow them to stay logged in whilst they browse the website. Again, the server will user PHP to carry this out. The session variables will also be reset - using PHP - when the user logs out. When the user update their details, the server will update them in the table in the database using a combination of PHP and SQL queries.

After the user logs in the server will post a personalised welcome message to them on the homepage using PHP and the session variable sent when they logged in. When the user posts a message the server will store it in the database along with the users details. The

messages that are stored will be compiled into a table that all users can view. The server will again use PHP to complete these tasks and a SQL query to store the message in the database.

The validation of all data entries will be carried out using JavaScript and also defined HTML commands.

Outputs

Registration

When filling in the registration form, should the user leave any sections blank an error message will display instructing them on exactly what has to be filled in. There is also certain criteria that some inputs have to meet such as: they must select a gender; the email address must contain an @ and a .co, and the password must contain a number, a lowercase character and an uppercase character and be between 8-20 characters. Any of these criteria not being met should result in an error message of some sort being displayed.

Login

If the user should enter a non-existent username, an existing username but with an incorrect password, or nothing at all then the server will return some kind of error message instructing them what they have done wrong and how to correct it.

Homepage

The first piece of user generated output will be a message on the homepage. When the user logs in to a registered account, they will be redirected to the homepage. On this page there will be information about the website along with a personalised message containing their username.

Message Board

From the homepage the user will have a choice to go to the message board, or their profile. On the message board should they choose to post a message, they will see it on the board. They will also be able to view all messages on the board regardless of whether or not they post one.

Profile

Should they choose to go for the profile page, it will contain the information that they entered when they registered. This information can be updated and will change on their profile if they do so.

End Users

To be able to register for the site the user must be between 11-18 years old. Therefore the users will all be of high school student age. As it is a small scale development as well all students who register will be from my school, Clydeview Academy. The site will be easy to

navigate as the students will range from being novice users of computer systems to expert users.

Scope and Constraints

The project must be delivered by 21st March 2017 for marking. This gives me around 6 months to complete it. In terms of resources, everything is available for free and can be collected for use when needed.

The internal deliverables of the project will include:-

- Gantt Chart
- User survey
- Test plan
- Project plan
- Feasibility study

The external deliverables of the project will include:-

- Showing a posted message and all the replies to that message on the site (screen)
- A profile page for all registered users (screen)
- Photo gallery with all user submitted photos on one page in order of most recent (screen)
- Hyperlinks to other pages including, homepage, profile page, other profile pages, photo gallery, calendar, update page, login page (screen)
- Wireframes of individual pages (paper)
- Storyboard of each individual page (paper)

Test Plan

Various tests will have to be conducted in order to deem the final product a success. To test the various elements of the website I will have to conduct multiple tests, some which may seem the same, but will test different aspects of the validation. Each test will be stated along with which specific requirements will be tested. The data that shall be used in testing will also be stated too.

The first test that I shall carry out will comprehensively test the registration form. The first set of test data I shall attempt to register for the website using is as follows:

- First name: [blank]
- Surname: [blank]
- Gender: [blank]
- Date of Birth: [blank]
- Email: [blank]
- Username: [blank]
- Password: [blank]

In theory, this should return a huge error message instructing me to fill in every detail on the form. Upon being instructed to do this I will use this second set of test data:

- First name: Benjamin-Michael-John
- Surname: Manson-Oxlade-Chamberlain
- Gender: male
- Date of Birth: 08/07/1941
- Email: ben.mansonatgmail.com
- Username: BenjaminManson99
- Password: pass1

This data should once again return a small error message, but with different instructions to the first. It should instruct me to enter a valid email address and a valid password. The pre defined validation HTML allows should prevent: first name from going beyond Benjamin-Michae as it reaches the maximum characters; surname from going beyond Manson-Oxlade-Chambe as it reaches the maximum length; date of birth from even being selected as it is out with the range of specified dates, and username from going beyond BenjaminManson9 as it reaches the maximum length. The third and final set of data that will be submitted through the registration form will be as follows:

- First name: Ben
- Surname: Manson
- Gender: male
- Date of Birth: 08/07/1999
- Email: benjmanson87@gmail.com
- Username: benmanson
- Password: 4ForkHandles

This data should be accepted by the server and inserted to the database comprehensively testing the registration form and the server-side process of registering a new user. From this I will return to the login form and attempt to login using this account. The data that I will use to test the login form will be in four stages:

Stage One

- Username: [blank]
- Password: [blank]

This should return an error instructing the user to try again.

Stage Two

- Username: menbanson
- Password: 4ForkHandles

This should return some kind of error message that alerts the user that the username entered does not exist.

Stage Three

- Username: benmanson
- Password: 4forkhandles

This should return an error alerting the user that the password they have entered is incorrect.

Stage Four

- Username: benmanson
- Password: 4ForkHandles

This final data set should log the user in to the site and allow them to browse whilst logged in to the 'benmanson' account. Once logged in, I shall navigate my way to the message board page. From here the posting of the message will be comprehensively tested. The message that I attempt to post will be carried out in 2 stages. For the first message I shall attempt to post:

"This message exceeds the maximum length, This message exceeds the maximum length, This message exceeds the maximum length, This message exceeds the maximum length,"

As stated in the message, this message exceeds the maximum length. The HTML defined validation command 'maxlength' should prevent this message from even being entered into the box. If the message is unable to be entered I shall enter, "Hello world" which the system should accept and enter to the database. If the message has been inserted into the database, then it should appear in the message board along with the username 'benmanson' and the time that I enter the message.

To test the complaint form, I will simply enter two separate messages to submit. They will be exactly the same messages as used to test the post message page. The first one like on the message page should be rejected as it exceeds the maximum length, and the second one should be accepted and inserted to the complaint table with the userID matching the userID of the benmanson account.

Having tested the complaint form, only one element of user input will be left to test - the updating of a users details. To access this section of the website I will have navigate to the profile page. Just as a check when on the profile page the following details should be displayed:

- Name: Ben Manson
- Gender: male
- Date of Birth: 08/07/1999

On this page there will be hyperlinks to update pages for each of the details able to be updated. These are name, date of birth, gender, and password. These details will all be tested for multiple different inputs to make sure that all validation works correctly.

Name

Stage One

- First name: [blank]
- Surname: [blank]

This data should display an error message due to not submitting any data.

Stage Two

- First name: Benjamin
- Surname: Menson

Having submitted this name change, I will move on to test changing the date of birth.

Date of Birth

Stage One

- Date of birth: [blank]

This data should display an error message due to not submitting any data.

Stage Two

- Date of birth: 08/07/1941

This date is outside the limits and should therefore not be able to be entered.

Stage Three

- Date of birth: 08/07/1998

This date should be accepted by the page and should therefore update the date of birth in the database. Moving on from this I will test changing the gender.

Gender

Stage One

- Gender: [blank]

This data should display an error message due to not submitting any data.

Stage Two

- Gender: female

This gender should be accepted by the page and should therefore update the gender in the database. Moving on from this I will test changing the password.

Password

Stage One

- Password: window

This password is too short and therefore will not be accepted.

Stage Two

- Password: windowglow

This password is the correct length, but does not contain a capital letter or a number.

Stage Three

- Password: W1ndowGlow

This password is the correct length, and also contains a capital letter, a lowercase letter and a number and will therefore be accepted. The database should be checked to make sure that it has been updated correctly.

In order for the data to confirm the data has changed, I will then log out of the account and attempt to log back in using the new password - W1ndowGlow. This will test the functionality of the log out feature. The profile will then be checked to make sure that the name, date of birth and gender have also been updated on the profile as well as on the database.

For the final test of the website, I will test that the links in the navigation bar to make sure it is in working order. All other links on the site will be tested whilst navigating to other areas to test in the plan above. All test results will be recorded in a table that will be combined when it is time to test the solution.

Design

Data Dictionary

Table	Field	Data Type	Length	Key	Required	Auto Increment	Constraints
pupilDetails	userID	integer	11	PK	Y	Y	
pupilDetails	firstName	string	15		Y		
pupilDetails	surname	string	20		Y		
pupilDetails	gender	string	1		Y		m or f
pupilDetails	dateOfBirth	date			Y		year >1997 and <2006
pupilDetails	email	string	25		Y		
pupilDetails	username	string	15		Y		
pupilDetails	password	string	20		Y		
message	messageID	integer	11	PK	Y	Y	
message	userID	integer	11	FK	Y		Exist in pupilDetail
message	messageContent	string	140		Y		
message	messageDate	datetime			Y		
complaint	complaintID	integer	11	PK	Y	Y	
complaint	userID	integer	11	FK	Y		Exist in pupilDetail
complaint	complaintContent	integer	140		Y		
complaint	complaintDate	datetime					

SQL Queries

```
CREATE DATABASE studentNet;

CREATE TABLE userDetails
(
        userID INT NOT NULL AUTO_INCREMENT,
        firstName VARCHAR(15) NOT NULL,
        surname VARCHAR(20) NOT NULL,
        gender CHAR(1) NOT NULL,
        dateOfBirth DATE NOT NULL,
        email VARCHAR(25) NOT NULL,
```

```
        username VARCHAR(15) NOT NULL,
        password VARCHAR(20) NOT NULL,
        PRIMARY KEY (userID)
);

CREATE TABLE message
(
        messageID INT NOT NULL AUTO_INCREMENT,
        userID INT NOT NULL,
        messageContent VARCHAR(140) NOT NULL,
        messageDate DATE NOT NULL,
        PRIMARY KEY (messageID),
        FOREIGN KEY (userID) REFERENCES userDetails(userID)
);


CREATE TABLE complaint
(
        complaintID INT NOT NULL AUTO_INCREMENT,
        userID INT NOT NULL,
        complaintContent VARCHAR(140) NOT NULL,
        complaintDate DATE NOT NULL
        PRIMARY KEY (complaintID),
        FOREIGN KEY (userID) REFERENCES userDetails(userID)
);
```

These are the SQL queries for the design of the database. Other SQL queries will be included, but will be implemented into the PHP code.

Pseudocode

complaint.php

1. start session
2. assign server variables and connect to server
3. select the database
4. assign html form variables to PHP variables
5. write and implement SQL query to insert data to table
6. display confirmation of insert query
7. close link to database

config.php

1. assign server variables and connect to database

This will be implemented in certain individual pages, although this particular file will also be able to be accessed by multiple pages

homepage.php

1. start session

2. redirect to login page is not logged in
3. display welcome message using session variables

login.php

1. start session
2. assign server variables and connect to database
3. display error if failed connect
4. assign HTML form data to PHP variables
5. carry out SQL query to check if the username exists
6. display error if username doesn't exist
7. updates session variables to user details in PHP variables if username exists
8. display error message for incorrect password
9. close server link

logout.php

1. start session
2. destroy session, cleaning all session variables
3. redirect back to homepage

message.php

1. start session
2. assign server variables and connect to server
3. display error if failed to connect
4. select the database
5. assign database fields to PHP variables
6. create and carry out query to select all data from message table
7. display data in a table
8. close connection to database

post.php

1. start session
2. assign server variables and connect to server
3. display error if failed to connect
4. select the database
5. assign HTML form data to PHP variable
6. insert PHP variables to database using SQL query
7. redirect back to message board page
8. close server connection

profile.php

1. start session
2. display PHP session variables in HTML table

registration.php

1. assign server variables and connect to server
2. display error message if failed to connect
3. select the database
4. assign HTML form data to PHP variables
5. create query to insert PHP variables to userDetails table
6. check whether existing username
7. if not then display success message
8. else display username taken error
9. close server connection

session.php

1. start session
2. link to config.php
3. assign session variables form login to PHP variables for use in other pages
4. redirect back to login page if no session variables set

NOTE: In all pages that begin with 1. start session, the page will link to the session.php file and use the session variables assigned there.

The following will be the pseudocode for one update page. All other update pages will follow the exact same process, but for different pieces of data.

updatePart1.php

1. start session
2. display HTML form, but with username pre-assigned from session variables

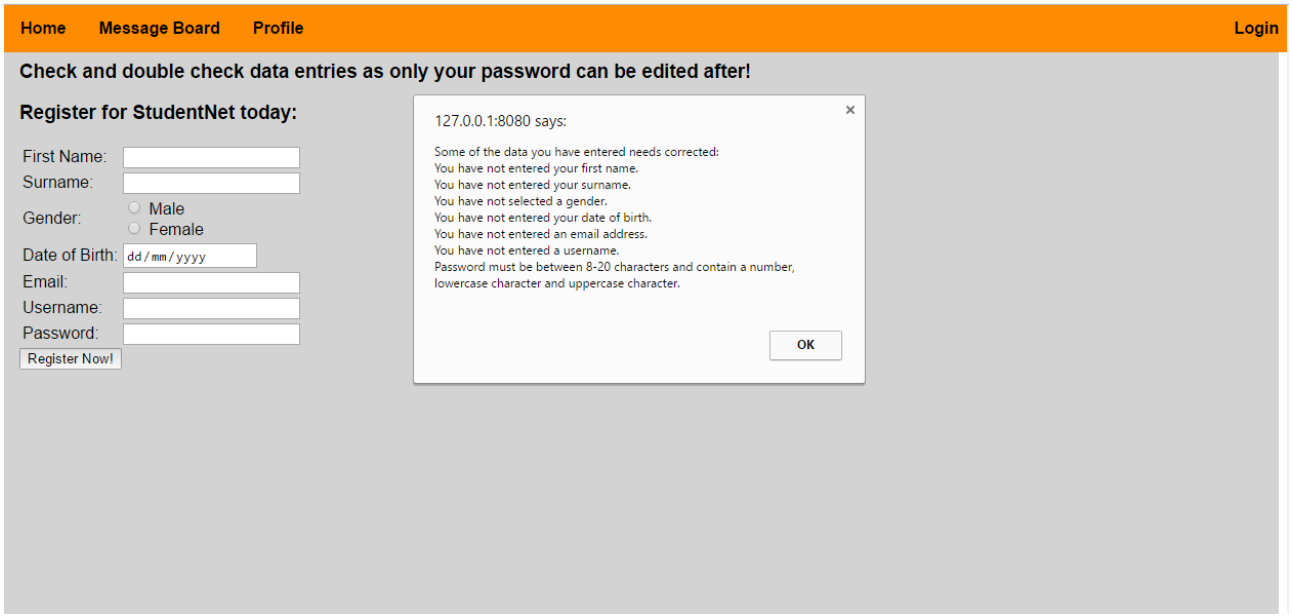
updatePart2.php

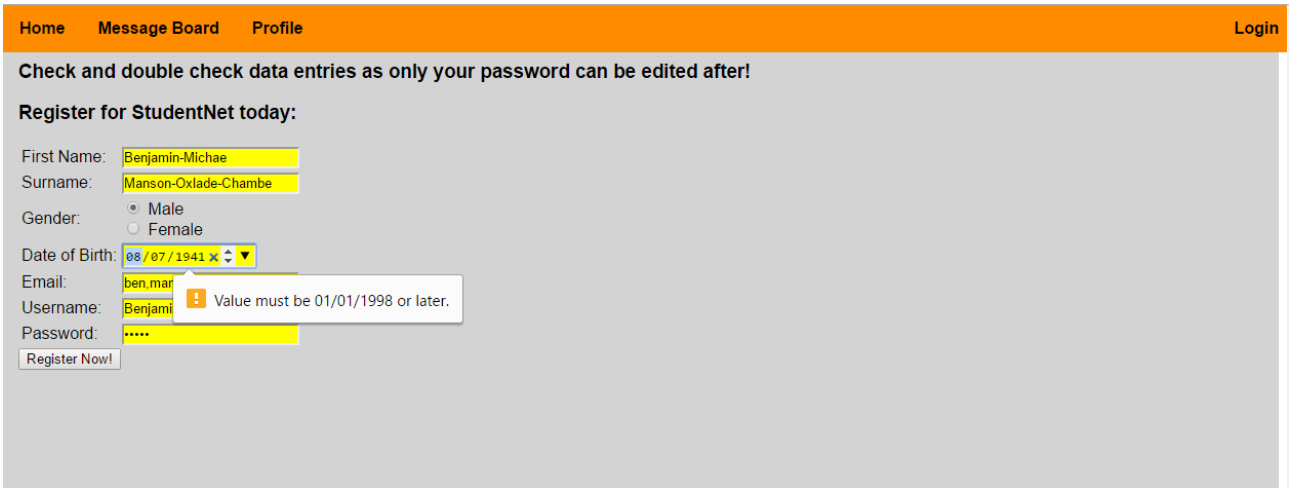
1. start session
2. assign server variables and connect to server
3. display error if failed to connect
4. select the database
5. assign HTML form data to PHP variables
6. create and carry out query to update data in database to PHP values
7. display success message
8. close the connection

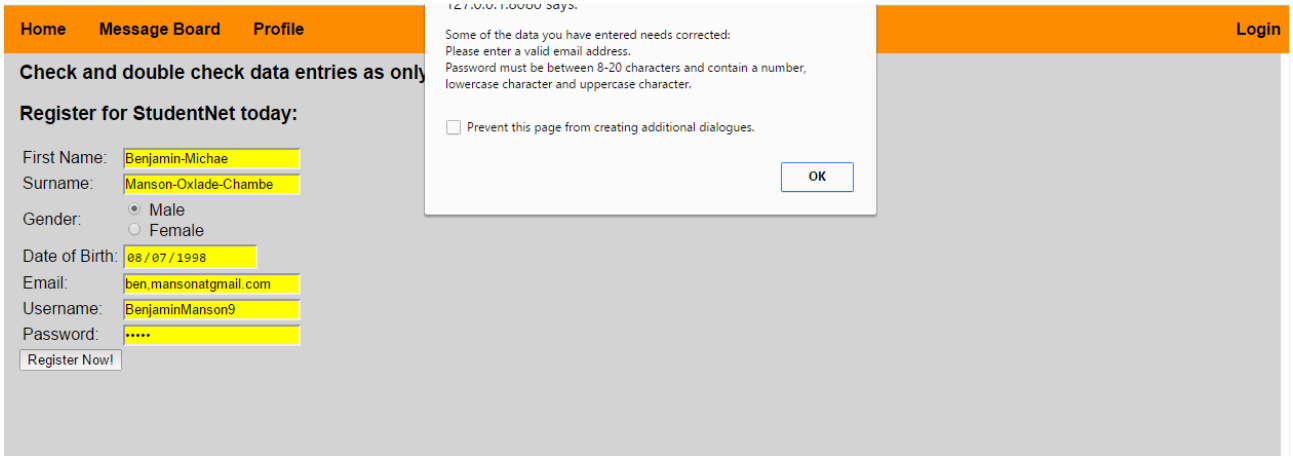
Testing

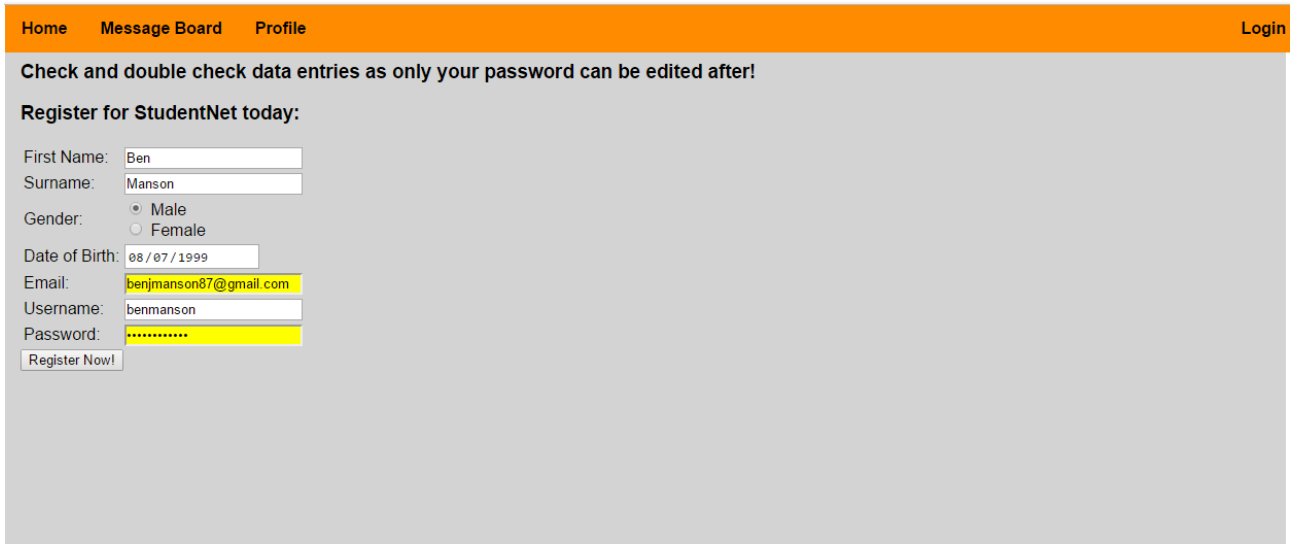
Having implemented all of the requirements from the requirements specification, comprehensive testing will be carried out using the test plan to test every element of the solution. The first element in the test plan to be tested is the registration form. This test report will include screenshots of evidence of the testing. The data, test and expected output are in the testing table included and this should be used for reference.

Registration









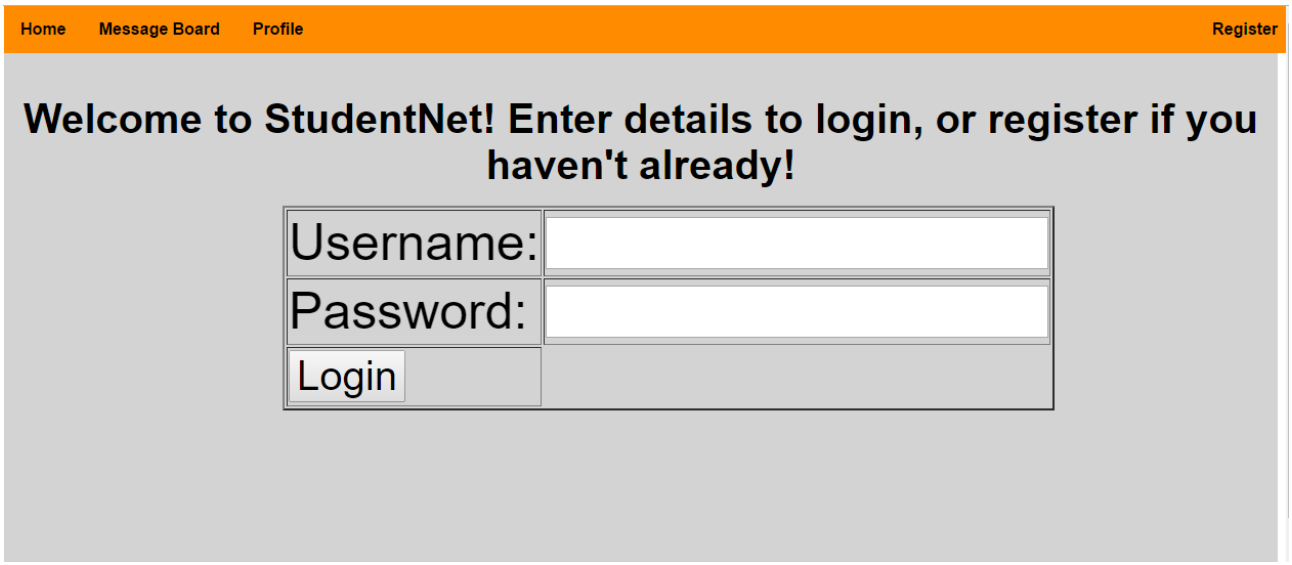
+ Options

	userID	firstName	surname	gender	dateOfBirth	email	username	password
Edit Copy Delete	12	Test	Male	m	1999-01-01	testmale	testmale	test
Edit Copy Delete	13	Test	Female	f	1999-01-01	testfemale	testfemale	test
Edit Copy Delete	86	Ben	Manson	m	1999-07-08	benjmanson87@gmail.com	benmanson	4ForkHandles

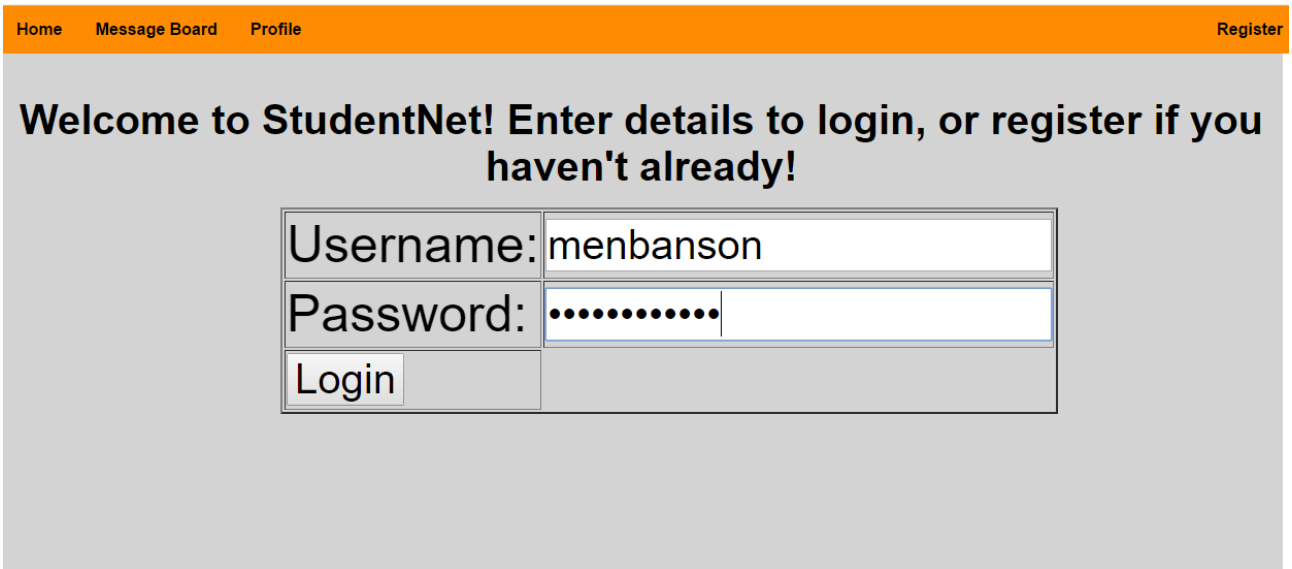
Check All *With selected:* Change Delete Export

Ben Manson

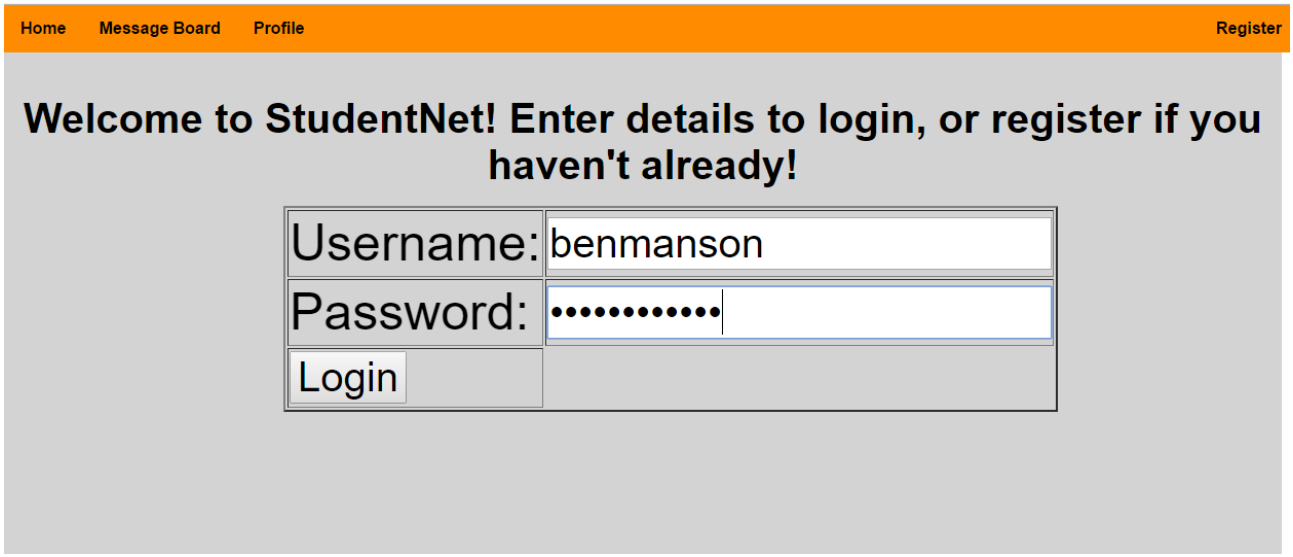
Login


Home
Message Board
Profile
Register
Welcome to StudentNet! Enter details to login, or register if you haven't already!
Username:
Password:
Login

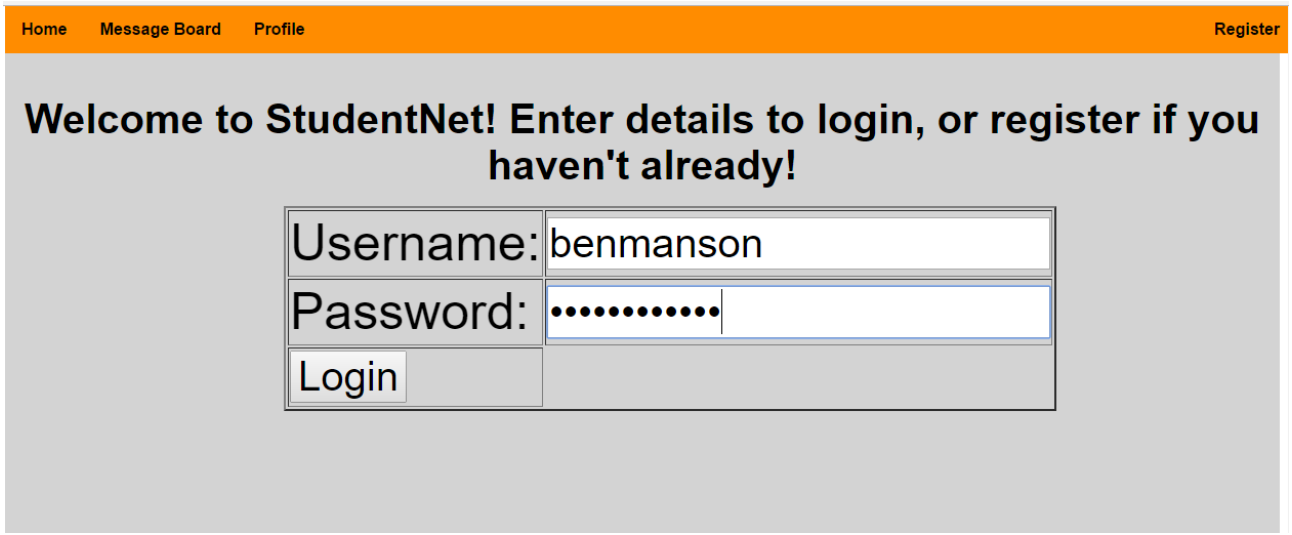

Home
Message Board
Profile
Login
Sorry, username is unregistered.
Click here to register

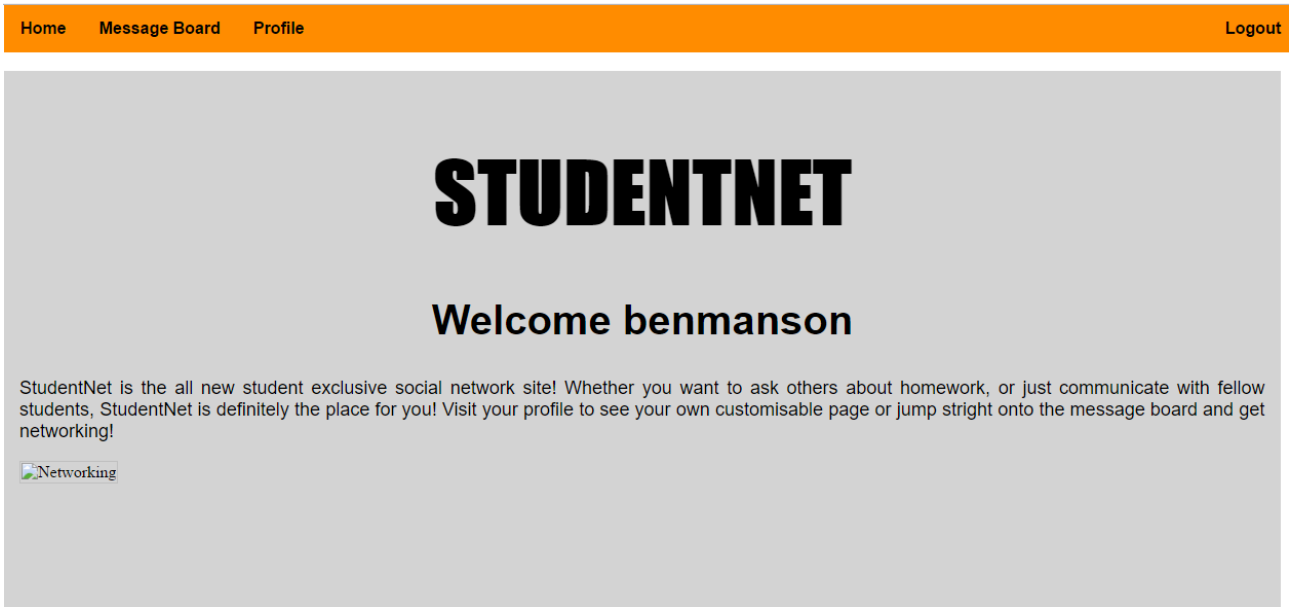

Home
Message Board
Profile
Register
Welcome to StudentNet! Enter details to login, or register if you haven't already!
Username:
menbanson
Password:
Login


Home
Message Board
Profile
Login
Sorry, username is unregistered.
Click here to register

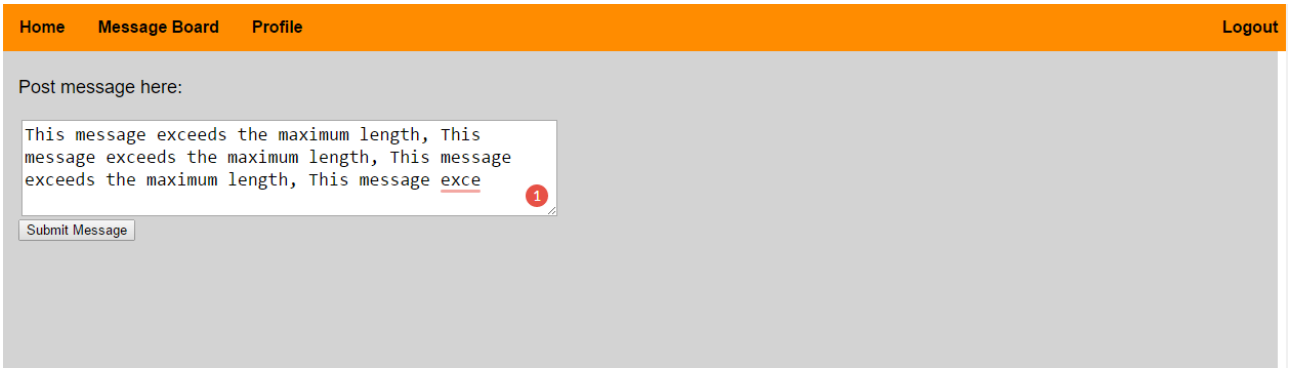

Home
Message Board
Profile
Register
Welcome to StudentNet! Enter details to login, or register if you haven't already!
Username:
benmanson
Password:
Login


Home
Message Board
Profile
Login
Incorrect password. Click here to return to Login Page



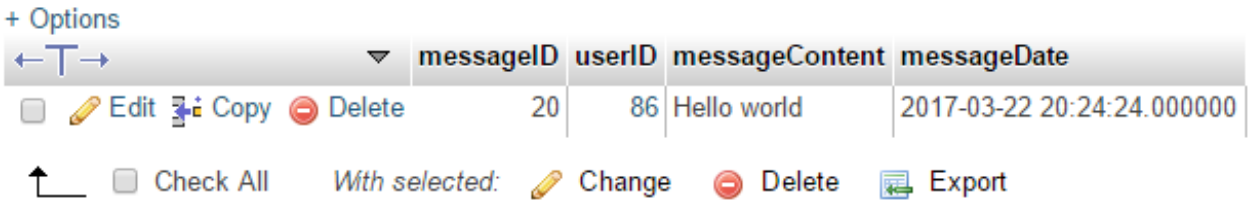


Message Board

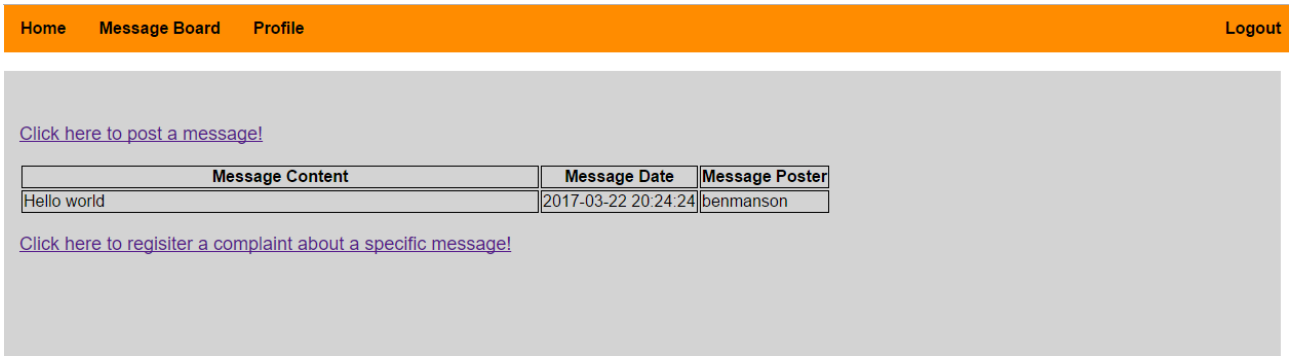


Message length is too long and therefore message cuts off mid way through.

Ben Manson



Message inserted into table.

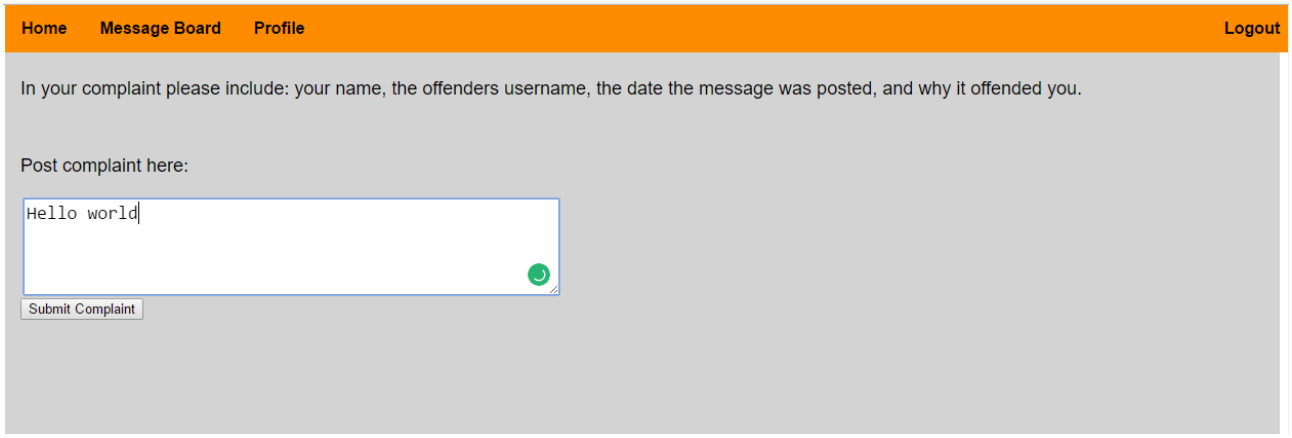


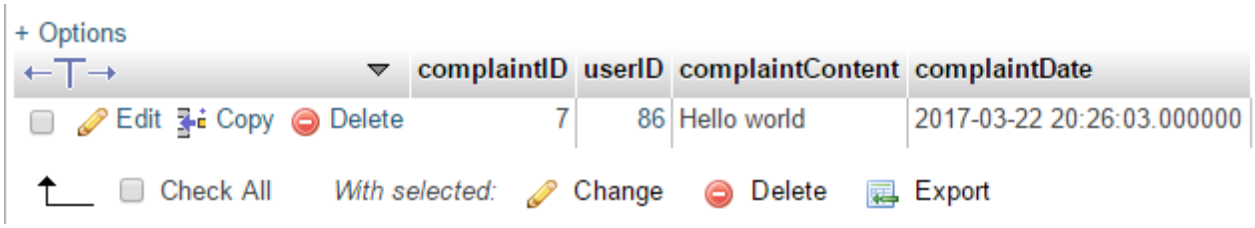
Message displayed on message board.

Complaint

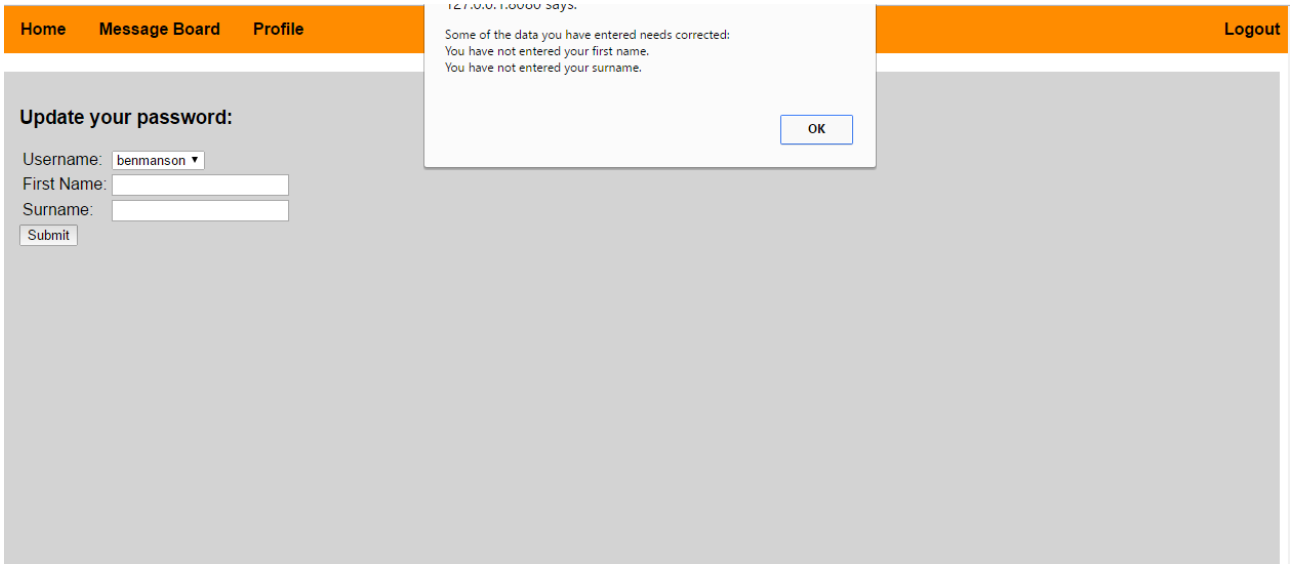
Ben Manson

Message cuts off mid way through as it is too long.

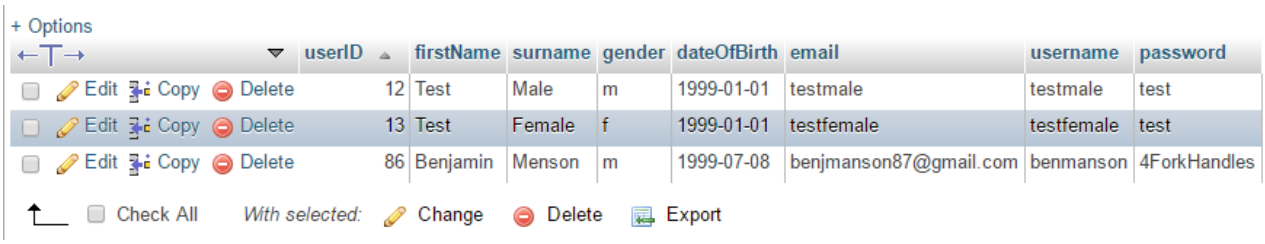




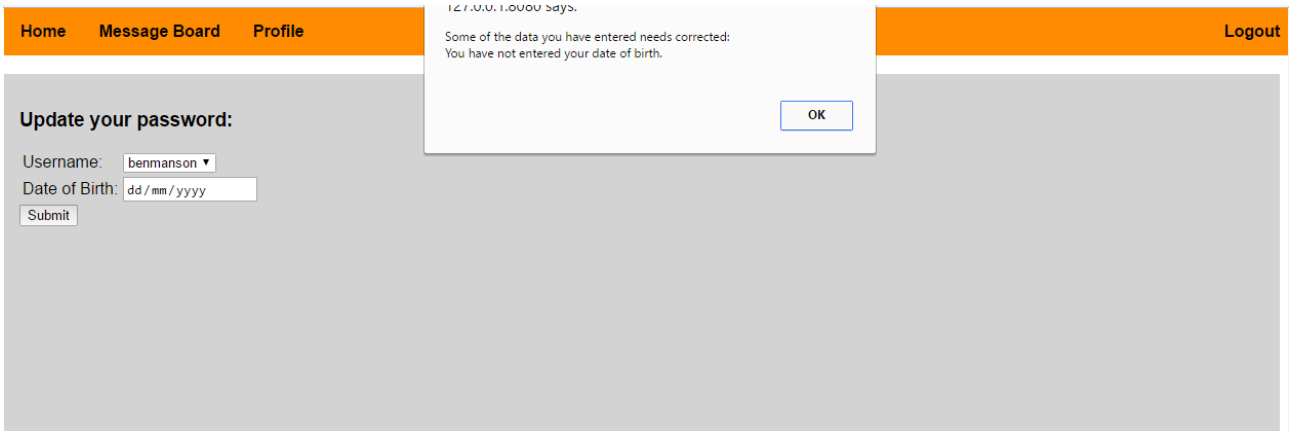
Update Name



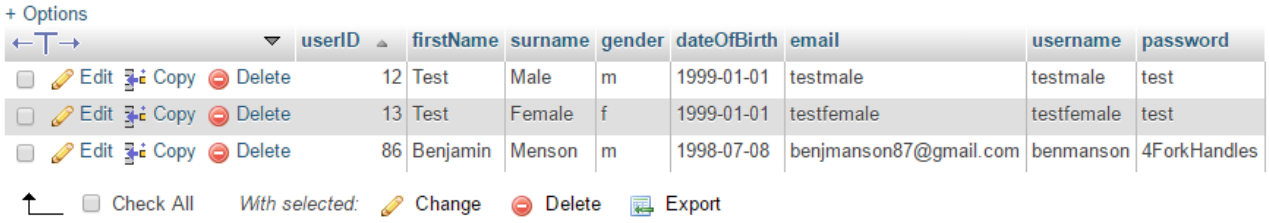
Ben Manson



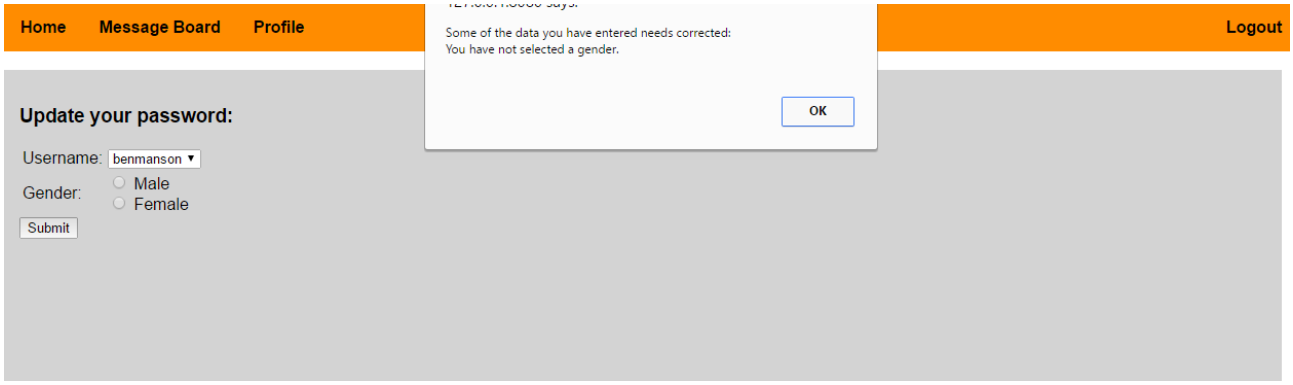
Update Date of Birth

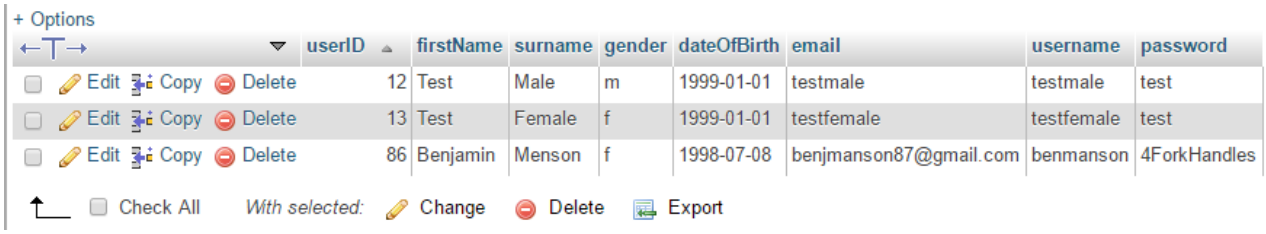


Ben Manson



Update Gender



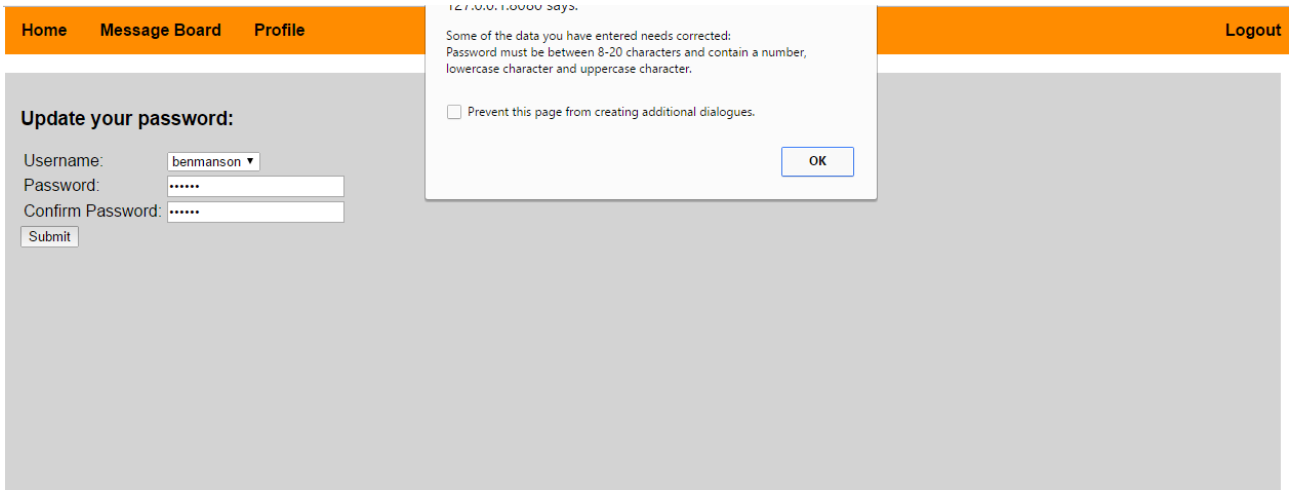


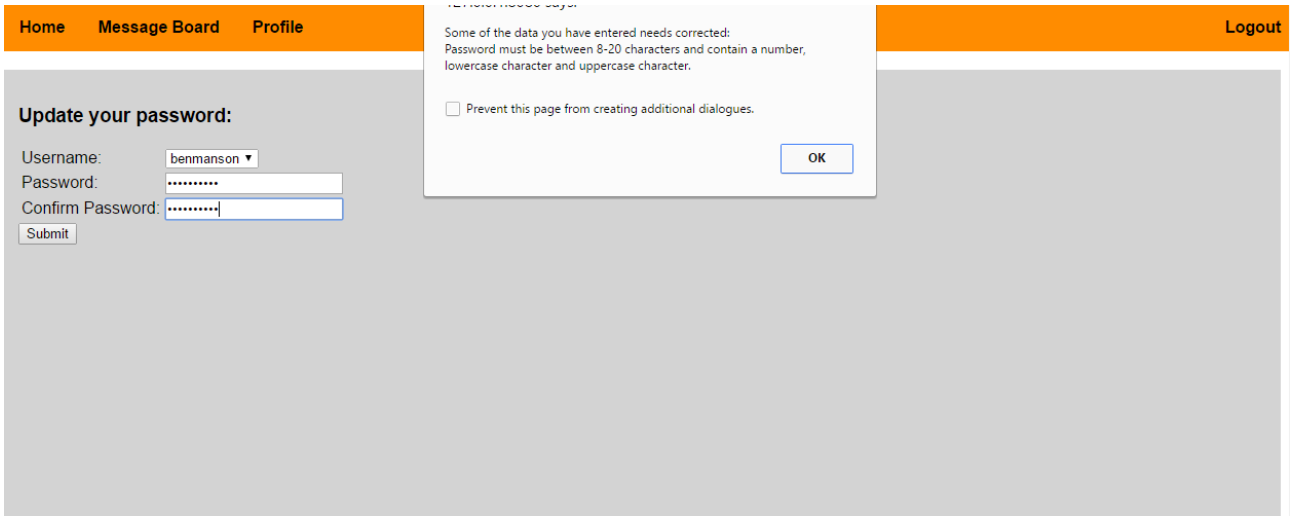
+ Options

	userID	firstName	surname	gender	dateOfBirth	email	username	password
Edit Copy Delete	12	Test	Male	m	1999-01-01	testmale	testmale	test
Edit Copy Delete	13	Test	Female	f	1999-01-01	testfemale	testfemale	test
Edit Copy Delete	86	Benjamin	Menson	f	1998-07-08	benjmanson87@gmail.com	benmanson	4ForkHandles

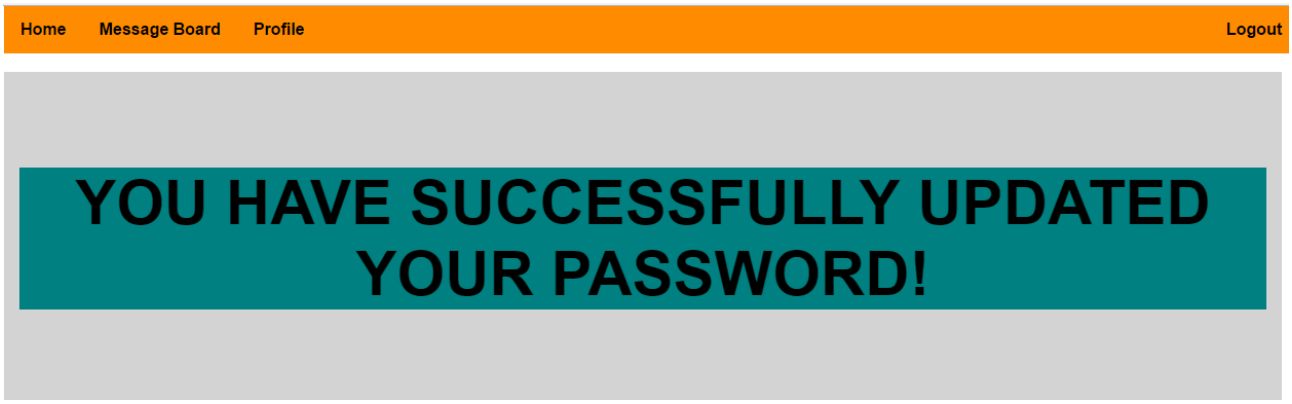
Check All *With selected:* Change Delete Export

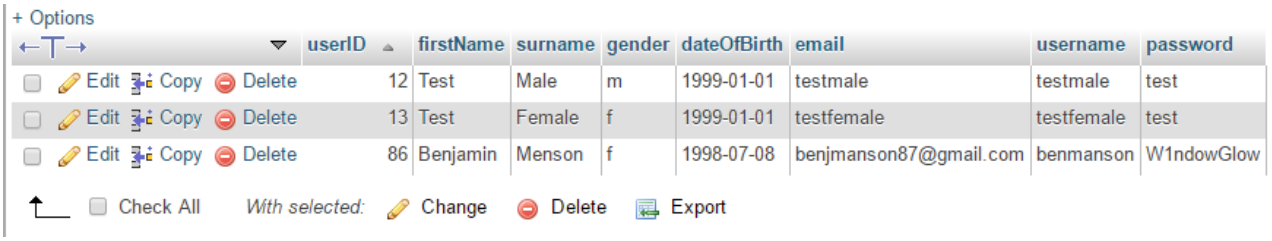
Update Password





Ben Manson





The above screenshots show all elements of the test plan have been fully and comprehensively tested. This confirms that the solution meets all requirements of the requirements specification and is therefore an acceptable, working solution.

Evaluation

The results of the testing all matched the expected results. This means that every single element of the site works exactly as specified in the requirements specification. After the comprehensive testing of the site I carried out very small scale user testing to see that the end users were able to use the site without any hassle. From the results of the user testing it can be concluded that the site is very easy to navigate. It can also be concluded that all elements of the site that worked during my own testing never caused any trouble during their testing. All of this can be used to finally conclude that the product is of a high enough standard to be released and fully matches the requirements that were set out at the beginning of the project.

In terms of my own performance, the time that I allocated myself for all the tasks proved to be too little. My implementation took me well over the 50 hours allocated to complete due to my severe lack of knowledge in server-side coding. This caused me to take around 60 hours for the implementation as well as several hours of building up my skills in PHP using various different web resources and forums. Due to taking up more hours in the implementation than I had planned, this meant that time for the paperwork after was limited. However, this has not proved to be too much of a problem after all. All problems encountered are mentioned in my progress diary and ongoing testing was carried out throughout the development with notes on problems encountered recorded in the diary also.

Overall, the project is up to the standard that I had intended it to be. Going into the project I had plans to add features such as a photo gallery and a calendar for students to add there own reminders to. However, I had not anticipated just how difficult these features would be to implement and therefore decided to not even include them in the planning stage. Another key feature that was left out was the feature of being able to view other user's profiles and add them to a contacts book. Again, during research before the project began, I discovered that I simply did not have enough time and manpower to implement such features and still complete the project in the time allocated. These features could possibly be designed and created in a future iteration of the design and development stages of the project. However, the site does match the requirements included in the requirements specification and certainly meets the Advanced Higher criteria that I had intended it to and therefore I would consider the development a success.