|  |  |
| --- | --- |
| A picture of a winding road and trees  Assignment 1\_IT7320  Student Result Support System | Abstract  An online system to help the student to check the result.  Yuhan Wang\_2152826 Desheng Liu\_2132346  IT7320 |

Table of Contents

[Summary 3](#_Toc521680062)

[Planning 4](#_Toc521680063)

[Requirements 10](#_Toc521680064)

[Tools Requirements 11](#_Toc521680065)

[Technique Requirements 12](#_Toc521680066)

[Testing Plan 13](#_Toc521680067)

[Testing Implementation 15](#_Toc521680068)

[1. Register page missed input 15](#_Toc521680069)

[2. Two passwords not match 16](#_Toc521680070)

[3. Login page missed input 17](#_Toc521680071)

[4. Successful login 18](#_Toc521680072)

[5. Register value store in database 19](#_Toc521680073)

[6. UI (User Interface) Test 20](#_Toc521680074)

[Demonnstrate of version control, git & github features 21](#_Toc521680075)

[Soulution of conflict 21](#_Toc521680076)

[Cloning repositories 23](#_Toc521680077)

[Merge 24](#_Toc521680078)

[Conclusion 29](#_Toc521680079)

[Appendix A 31](#_Toc521680080)

[Source code: 31](#_Toc521680081)

[Register.php 31](#_Toc521680082)

[login.php: 33](#_Toc521680083)

[Index.php: 35](#_Toc521680084)

[Server.php: 39](#_Toc521680085)

[Style.css: 42](#_Toc521680086)

[errors.php: 45](#_Toc521680087)

[users.sql: 46](#_Toc521680088)

[Appendix B 48](#_Toc521680089)

# Summary

We want to create a system for every student to help them to check the result because when I did some research that I found some student result check systems are not transparent to use for students and intricacy to maintenance for the tutor operation team. So my teammate and I want to build a simple website to identify solve this problem for tutors and students also can help them improve the efficiency. My team is planning to use the HTML language and PHP technique to demonstrate how this system works. By the way, our online system also can use on the mobile browser for Android and iOS platform as well.

# Planning

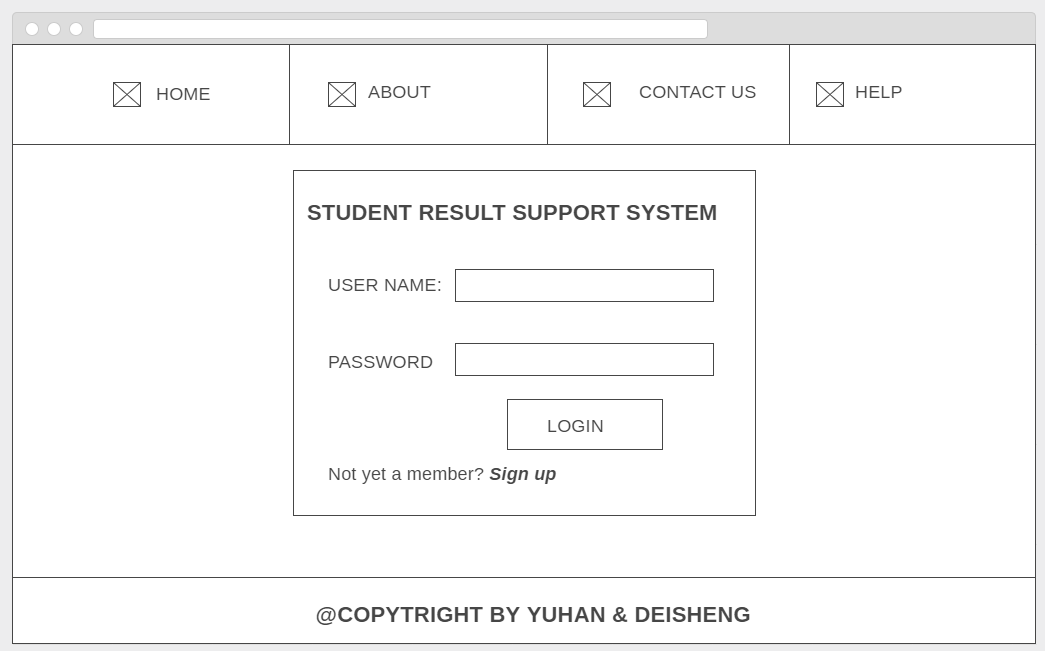
Firstly we will create a homepage that the student can log in straightly on this page. The design view looks like in figure 1-1. For the homepage is clear to use for every student, they only need to type their name and the password, and then they can log in. By the way, if the student does not have our system’s account, no worries the student can sign up the account when they click ‘sign up’ link under the login button. Then they can register an account for our system. For the techniques or methods, we will use the PHP language to implement the login function, we will check the username, and password with the database if all correct the student login success if not system prints the wrong message to alert. After the student login, we will use the session to store the student information, so the information can help the system find the courses they did. 

Figure 1-1

If the student click sign up button they will jump to registration page. In this page student need to set their username, password and also need to write the email address, because of if the student forget the password they can contact and we will ask them the original email address they wrote in this page, so email address is a backup for this situation. The design view looks like figure 1-2. The techniques or methods we will use is PHP to help us to implement this part. If the user inputs wrong password in ‘confirm password’ box the system will give the user an alert message to remind the student. Then if the student forget input username or email address the system also will print the alert message. When the student finish registration their information will be store in the database.

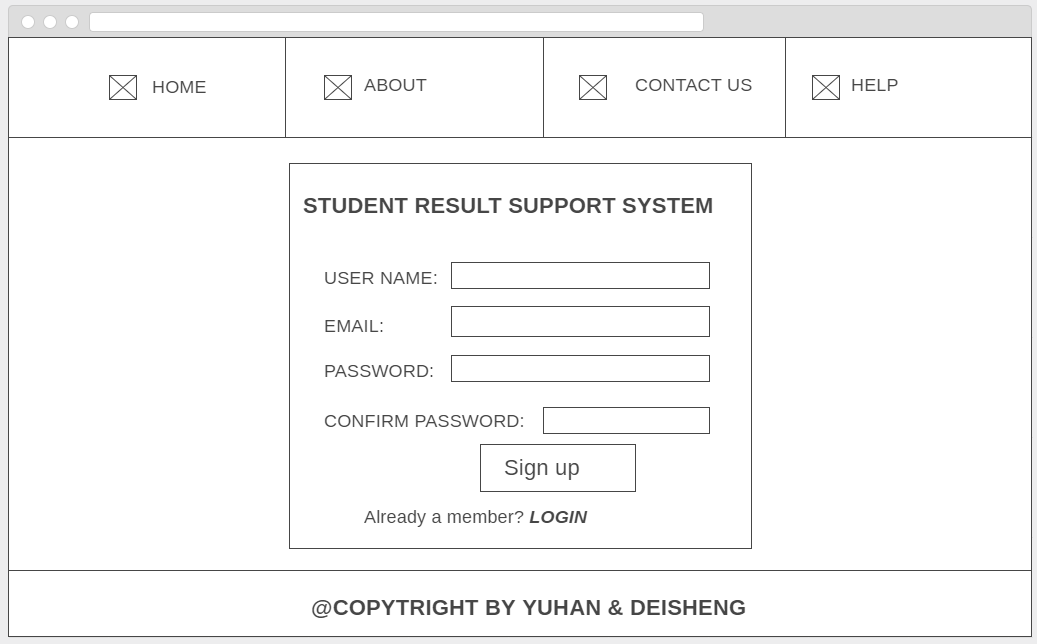


Figure 1-2

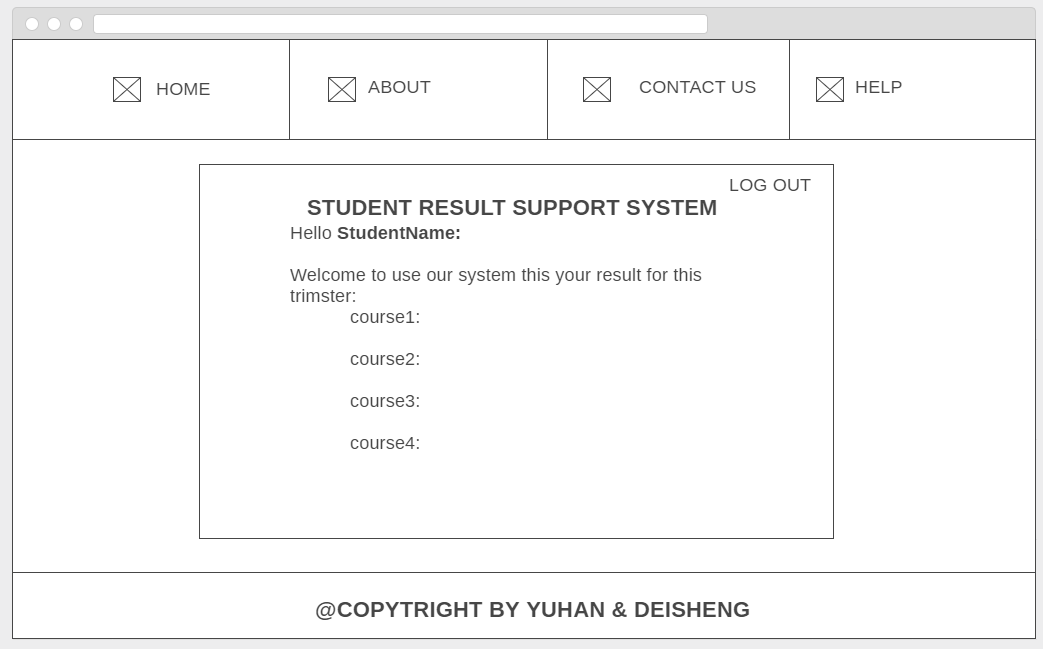
After the student login to the system, because of the session storage, this page will print the student’s name. The system will use the student name to find the courses and results for each student and print to implement those techniques and methods we will use PHP as well. So they can check the result of every course. The design view it looks like figure 1-3. In this page, the student can check the result when the tutor uploads into the database. If the student has any problems such as no result display, the wrong result, please go to CONTACT US page to email us (student.support@gmail.com). Finally, when you finished checking the result do not forget to log out to our system if you use the public computer.

Figure 1-3

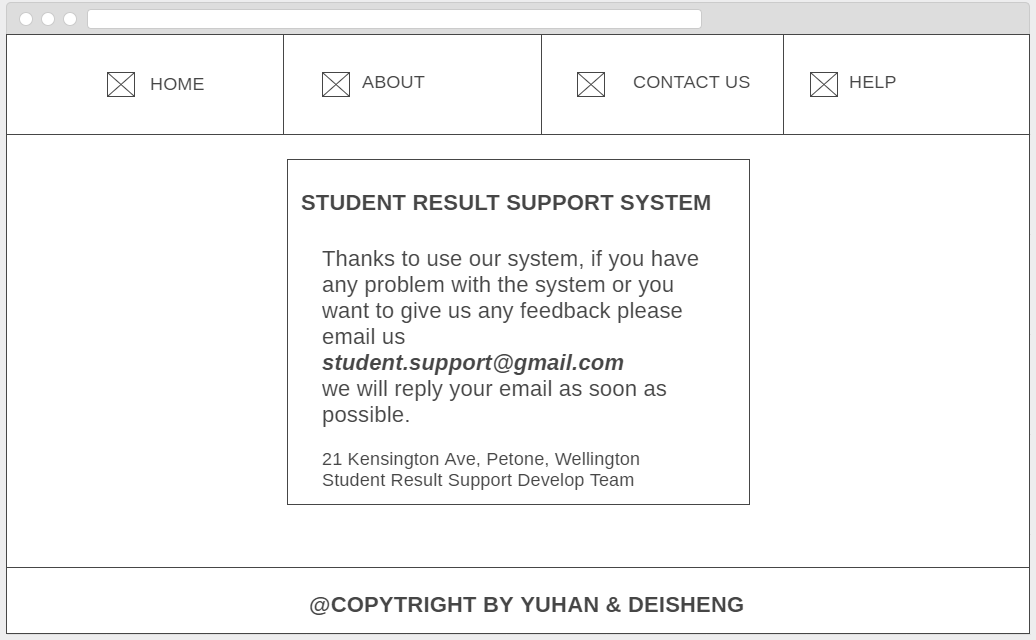
Any of students or tutor can find our support email address in contact us page. For this page, the design looks like figure 1-4. Because this is a straightforward online system for checking result, so I did not put any interesting elements or pictures on our website. However, we will use the bold and large font size for our email address. 

Figure 1-4

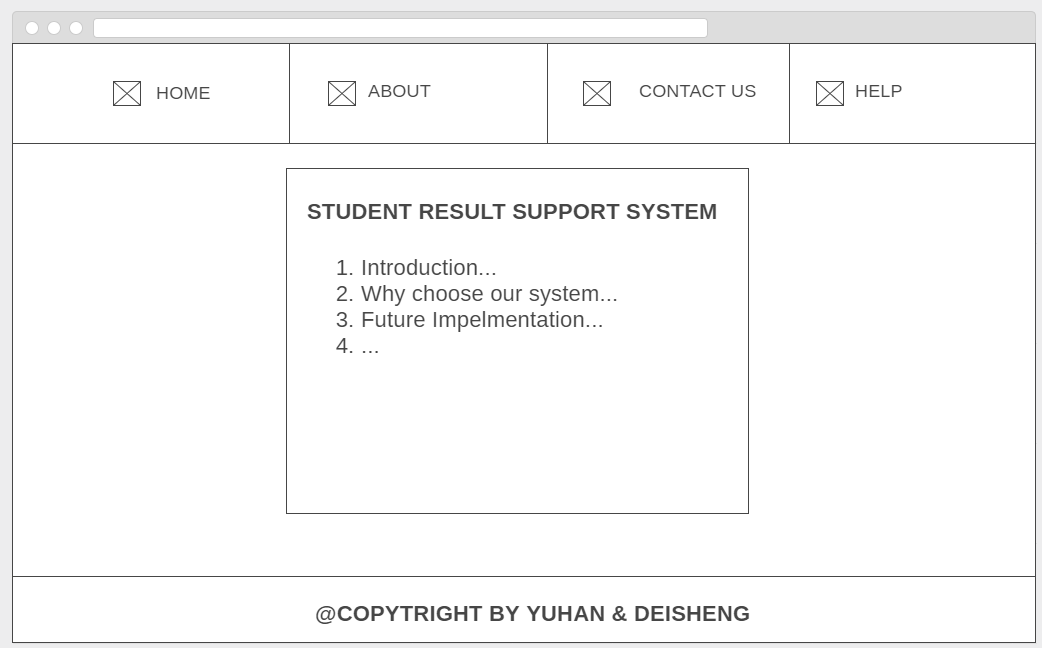
In about page, we want to do some introduction about our system. The important thing on this page is to tell the user why to choose our system and also talk about the future implementation of our system. For the design view it looks like figure 1-5, for the design of this page, I will use a different font size and probably different colors to demonstrate our information in this page.

Figure 1-5

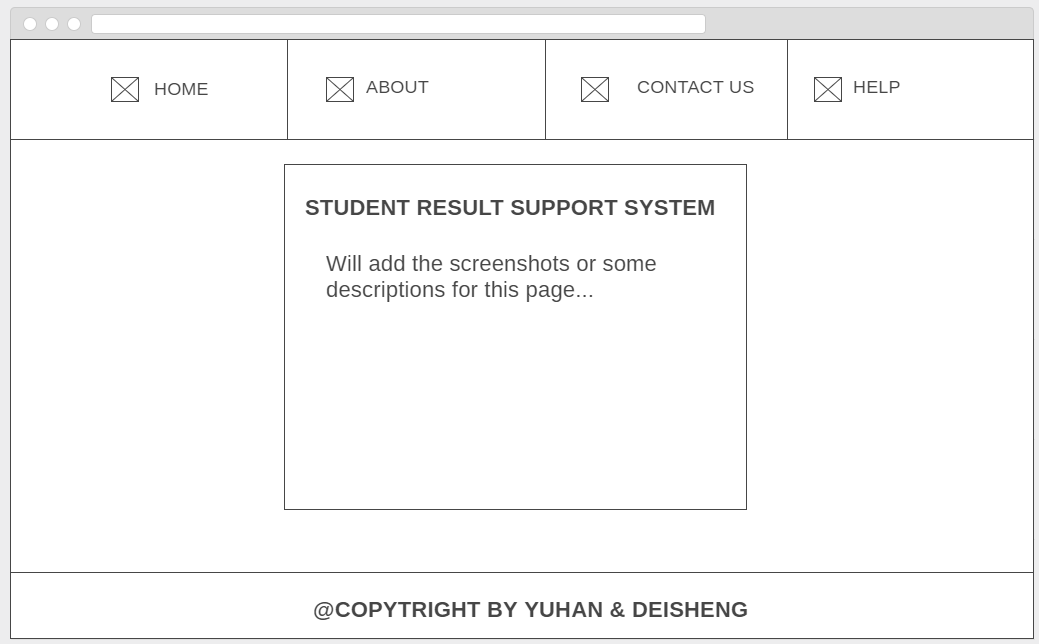
Finally is the help page, for this page our team are planning to add some screenshots and descriptions for our system, if the user has any problems they can look our help page first then if still cannot solve the problem, the user can contact our support department. The basic design view it looks like figure 1-6. 

Figure 1-6

# Requirements

The student result check system will be built online and can use in a most operating system for example windows, mac and also can use on mobile phone for Android, iOS platform.

In this system can implement student login and registration functions. For instance, the student uses our website they can log in to our system with the username and password if they do not have the account for the system they can register an account. All information from the students will be stored in the database. After a successful login to the system, the student can see the result for this trimester. The result for each course will be updated in the database by the administrator. The administrator has the privilege to use the database, depending on how many users to use the system, if we have more users in the future, we will add more administrator but for now, there are two administrators can have the database’s privilege. The school tutor and administrator will provide all the result will be put the data in the database. When the student finished using the system, we provide the logout function to clear current login student to keep them private for everyone. This is the basic structure for this phase.

For the future implementation of the system will behave more functions such as attendance checker, book appointment and something else into our system.

# Tools Requirements

* Wampserver
* MySQL
* Notepad++
* Sublime Text
* Windows & Mac System
* Android & iOS platform
* Chrome, Firefox, IE…
* Github

# Technique Requirements

To build the system, we used the necessary CSS, HTML and JavaScript for the front-end. For the back-end, the most technique we used is PHP and MySQL. For connecting the database and HTML, we used PHP, when user login to the system we used session to store the username and display the username in the check result page then for student logout the system, the back-end will close the session to keep the private security level.

For database we will use MySQL to update the student result and then use PHP to print on result page.

# Testing Plan

Firstly before testing the student result check system, we need to find what to test first and what to test most. I used the Risk Matrix to demonstrate the risks of multi-dimensions that will be hidden in our system. Then I partition the risk in three levels High, Mid, Low (H, M, and L). From the form, I find there are two high lever risks in our system, knowledge of area and impact on users. Also has two mid-level risks high traffic and how visible. 

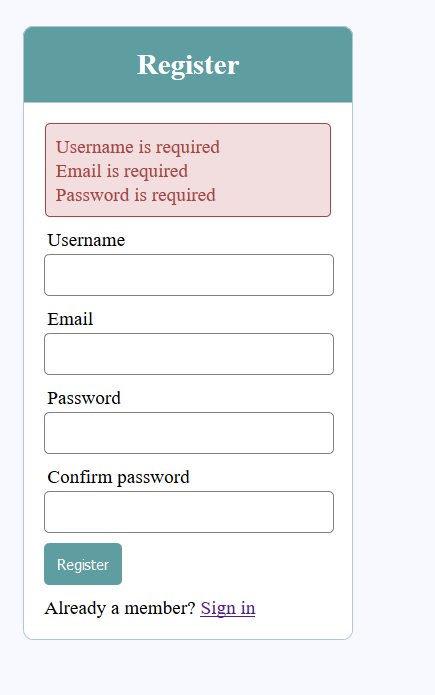
The knowledge of the area is the knowledge for developers. We need to understand how this system works does. This could be a high risk in during the development, so we need to test every developer to confirm they understand the logic of the system. If not assume that will be big trouble for the future implementation. In our development group only with two peoples and we all understand the logic for the system but I still put this for high risk because we not sure we will have enough knowledge control for future development. Another high-risk level impacts on the user, if our system has a bug or error it will impact lots of users because students will use the system we need to test the function, performance and user interface as well. We will check the back end, and front end first because this part will spend lots of time. In the test we need to confirm the student can log in the system and see the result correctly, also can log out. After that, we need to test the registration function for each student. Finally, is the student registration information can store in the database correctly. Another part should test in the database in the back end. Make sure the administrator can insert data without errors. User interface part we also will test, but the current user interface is only available for pc web browser, we need to check the online system can fit most popular browser that user often uses.

There are also have two mid-level risks from my analysis in the form. I think High Traffic is one of the mid-level risks. The High Risk in this form does not mean the transport that means the high usage rate for users. Lots of students will use the system to check the result. This is the high traffic for the system. If our system crash with a little bug with effect many students. From here we can see we need to force the testing on function, performance, the user interface in front-end and database in the back-end. Another mid-risk is how visible when the system has a bug if the bug is happening in the user interface the user will see the bug. This situation is not suitable for our project. If the bug is occurring in the back-end, the user cannot see the bug easily. This situation does not affect our reputation for our product.

All in all, this is the risk analysis depend on the risk matrix for the system. We also find some low-level risks but because the time management reason we cannot test all things, so we need to test the function, performance, user interface most and first because if those things are correct then the system can run correctly as well at the same time we also can keep a high usage rate for users. This is our basic testing plan for the current version of the system.

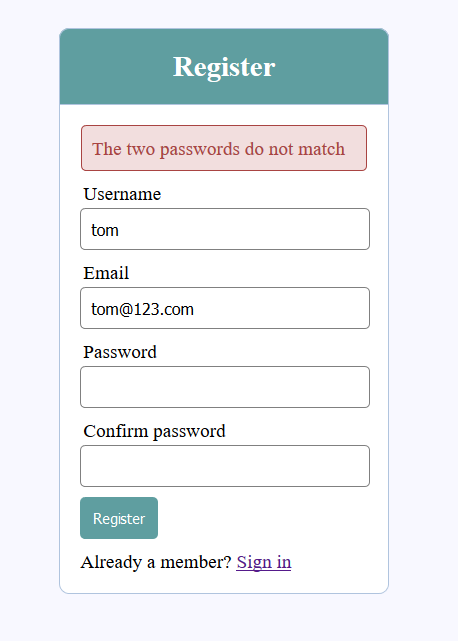
# Testing Implementation

## Register page missed input



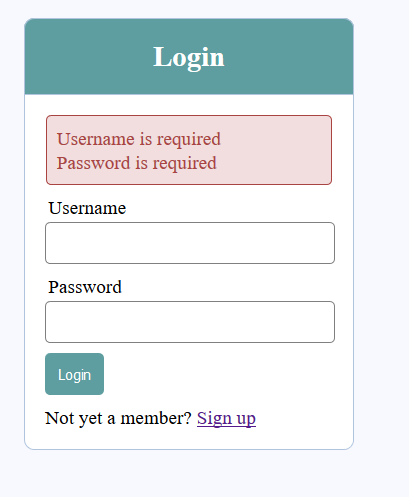
When user missed registration information, the system will show the error alert to reminder user what things they missed.

## Two passwords not match



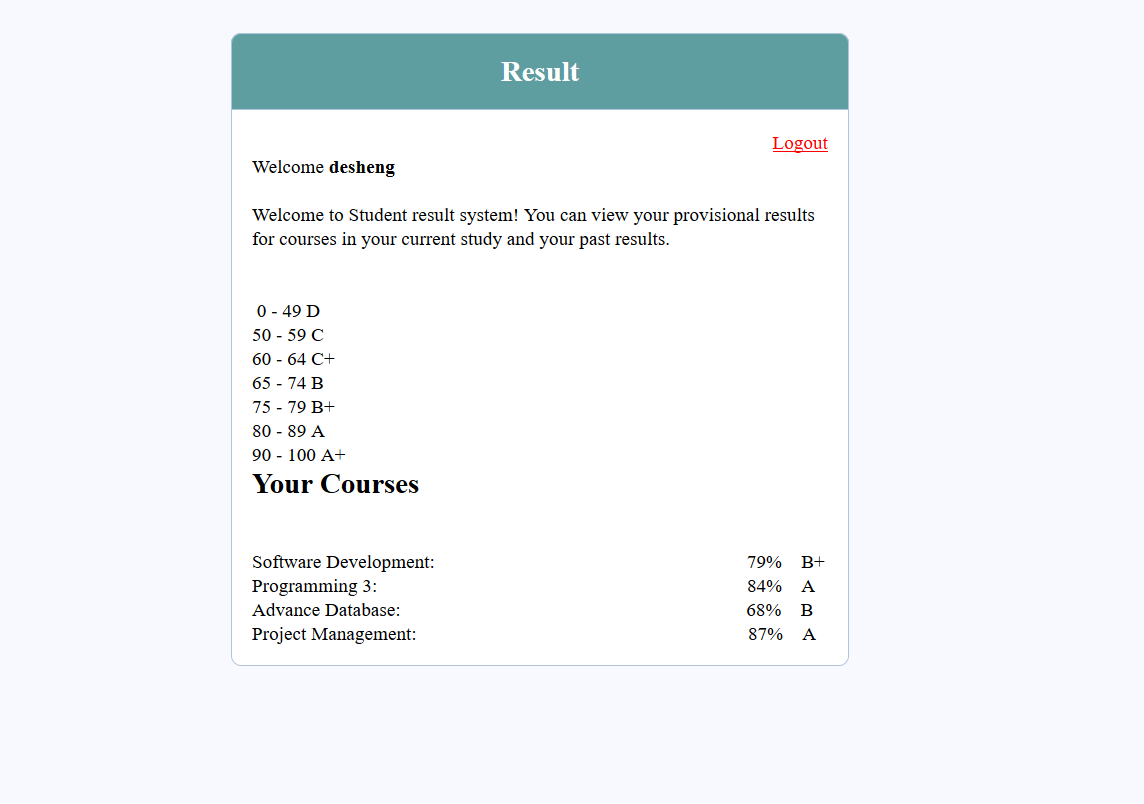
This situation might happen for lots of users, and sometimes the user types too fast then effect two passwords not match. The system will check and compare password and confirm the password. If two passwords are not same system will let the user know and print an error message.

## Login page missed input



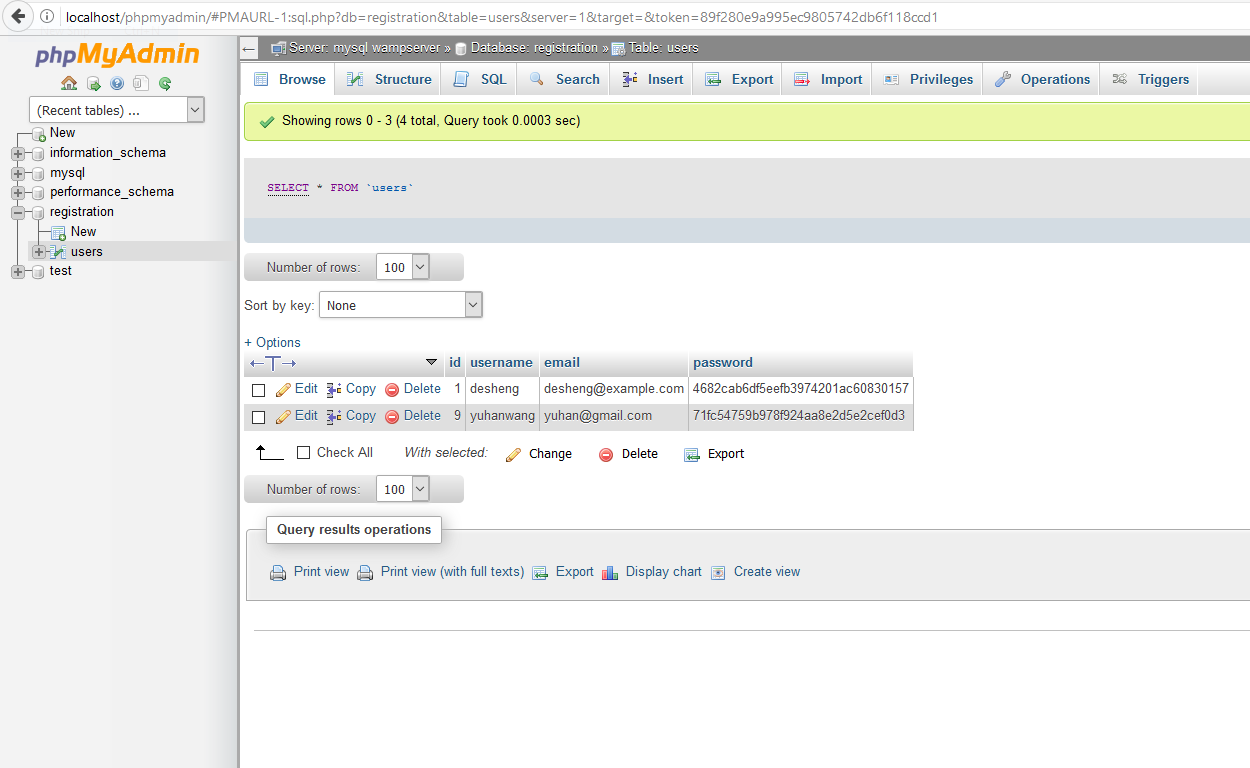
There is a function to test the username and password in the database or not if the password does not match the username student cannot log in to our system to check the result. The error message will print and a reminder user the input username or password wrong.

## Successful login



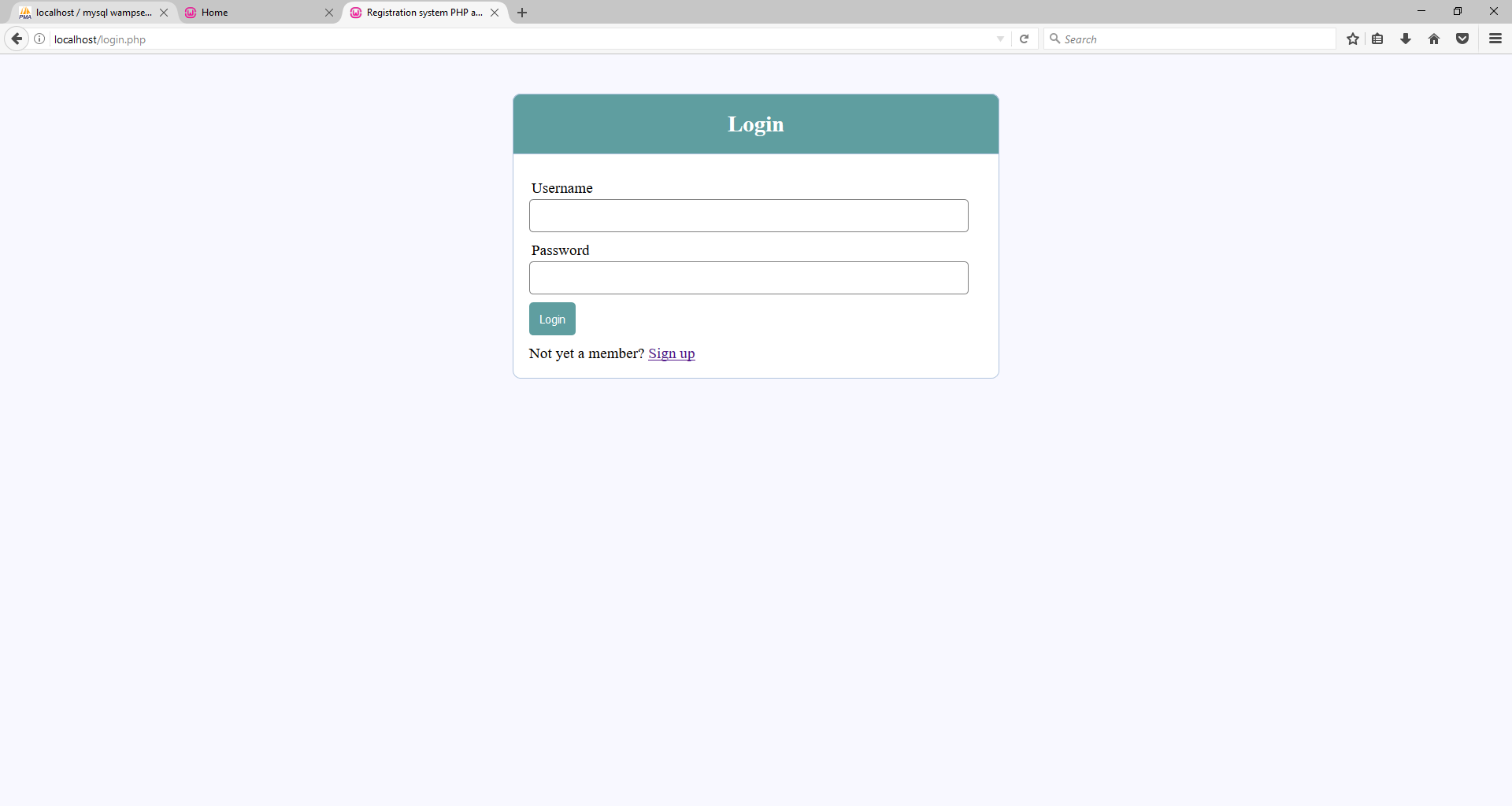
When the student log in to the system, they can see the result for their curse. The course and result are print correctly in this page.

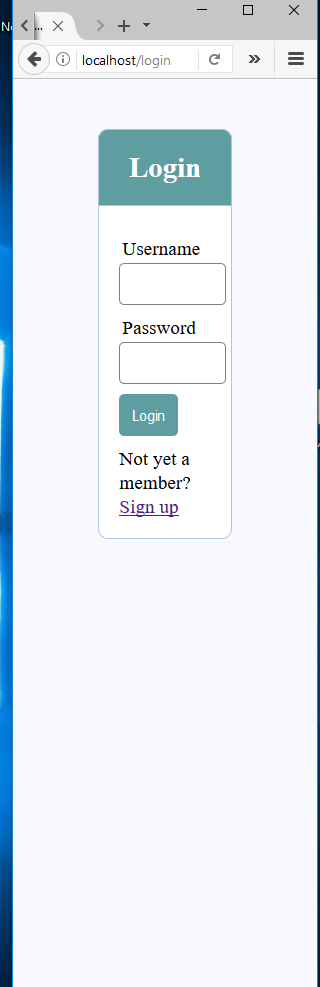
## Register value store in database



This test is to confirm the database can store the student information and use that information to check the student login. By the way, the passwords is encryption to keep the security level and the private for students.

## UI (User Interface) Test

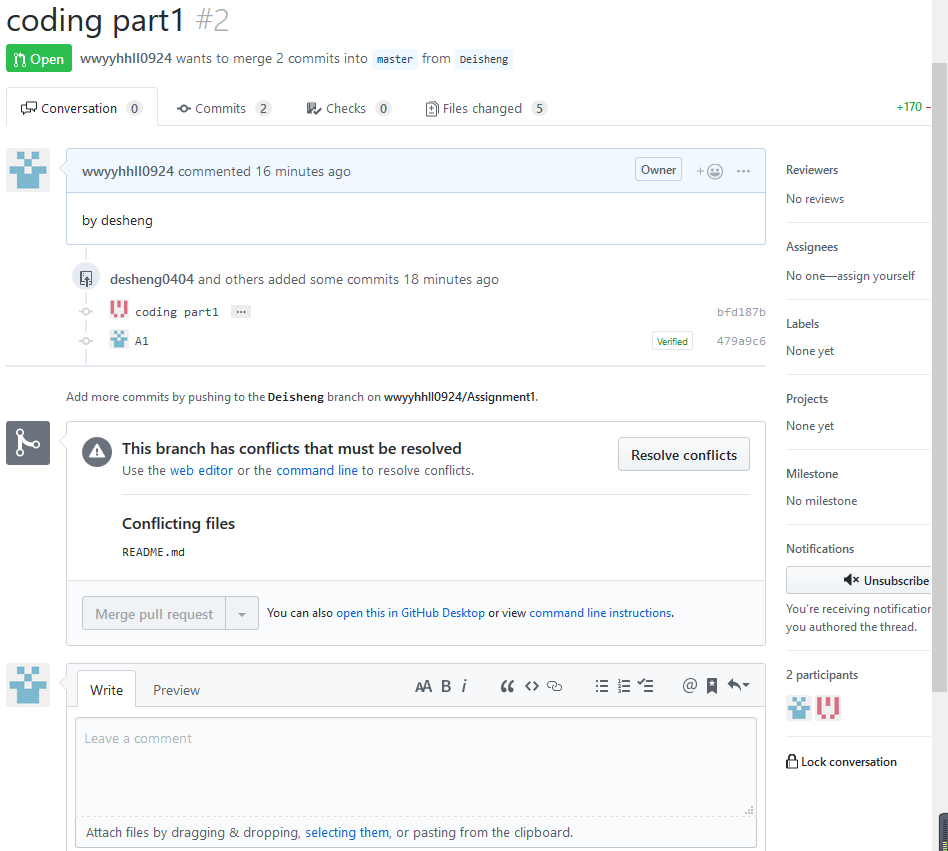


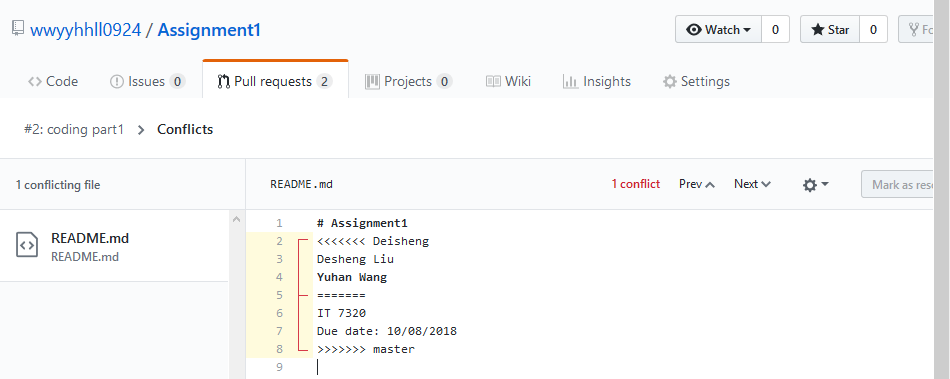


We test the different UI size for the browser to keep the control panel, not deformation, and it can fit for any browser with different size.

# Demonnstrate of version control, git & github features

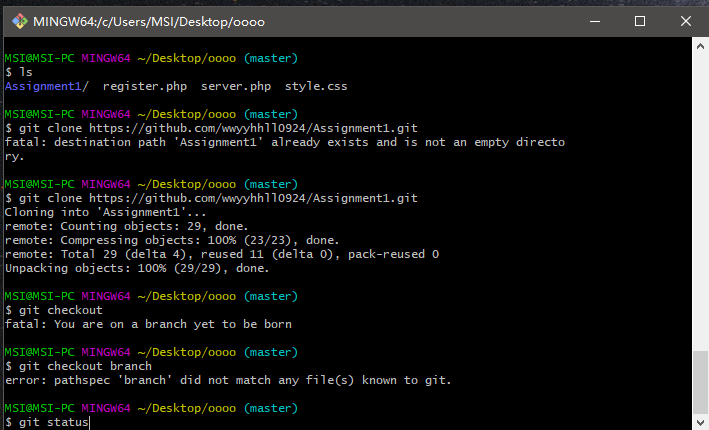
## Soulution of conflict

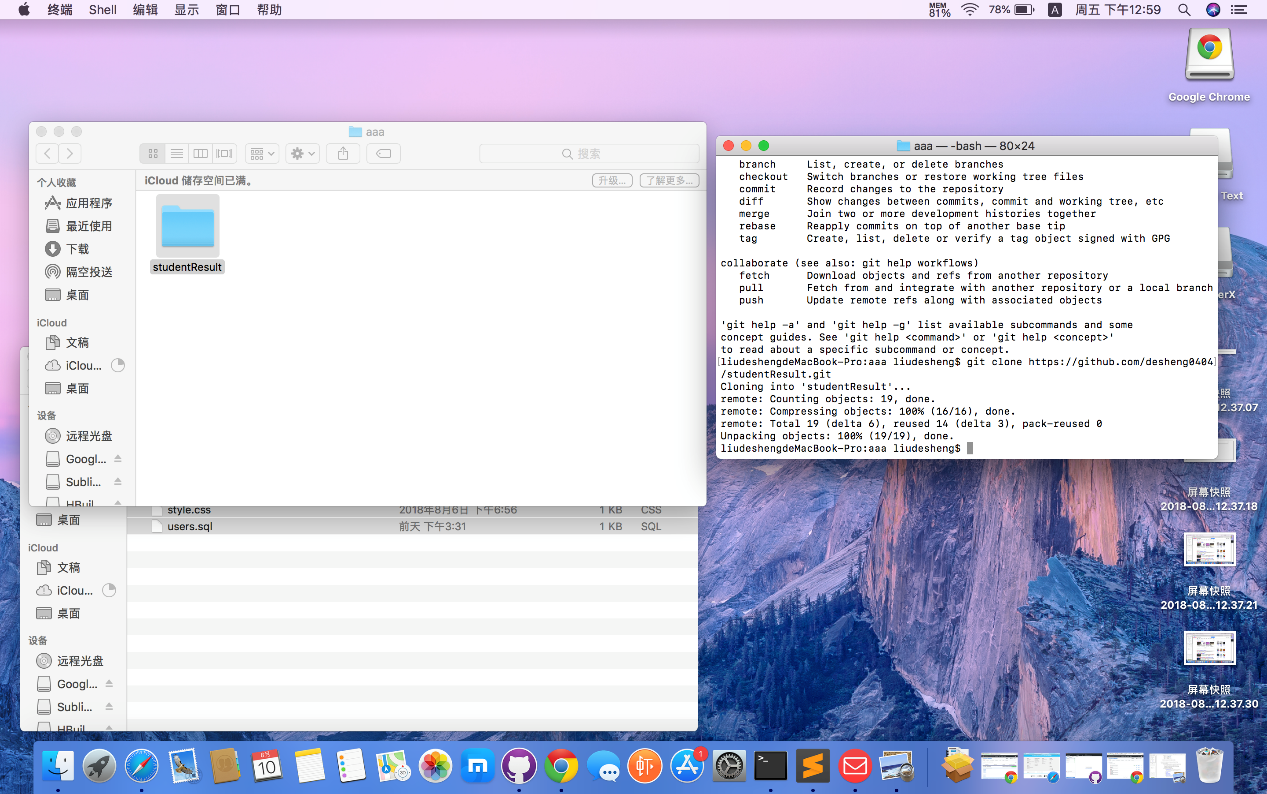




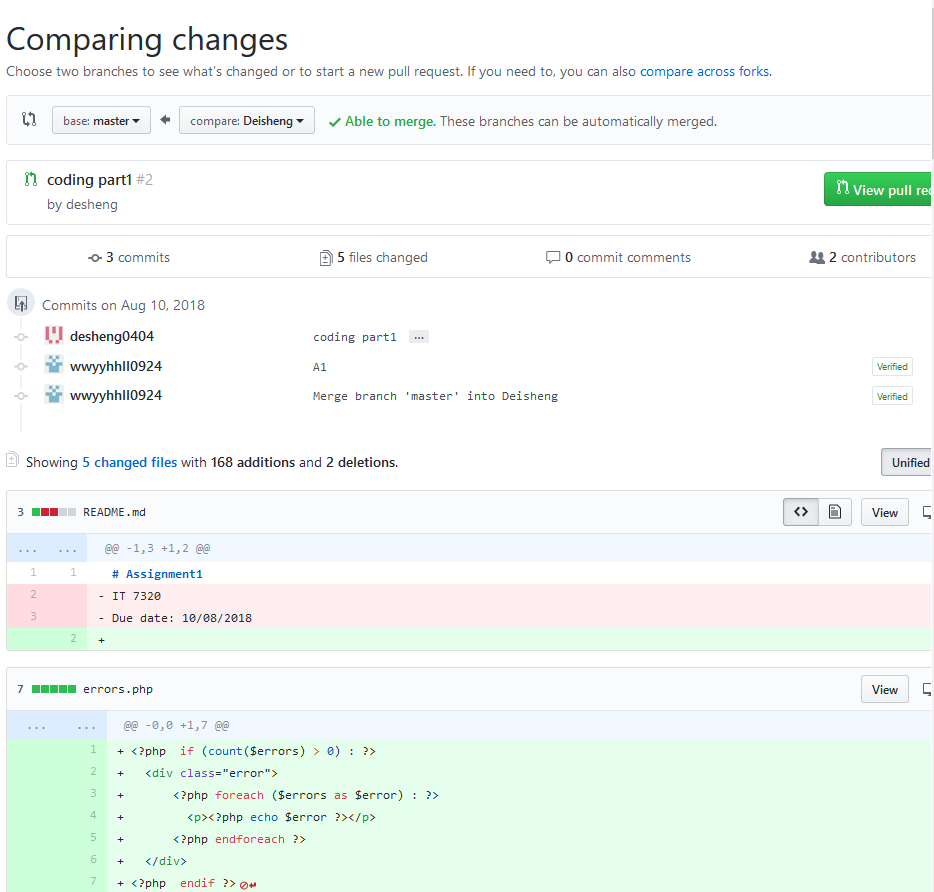
Delete and change the different contexts between master branch and deisheng branch for sync.

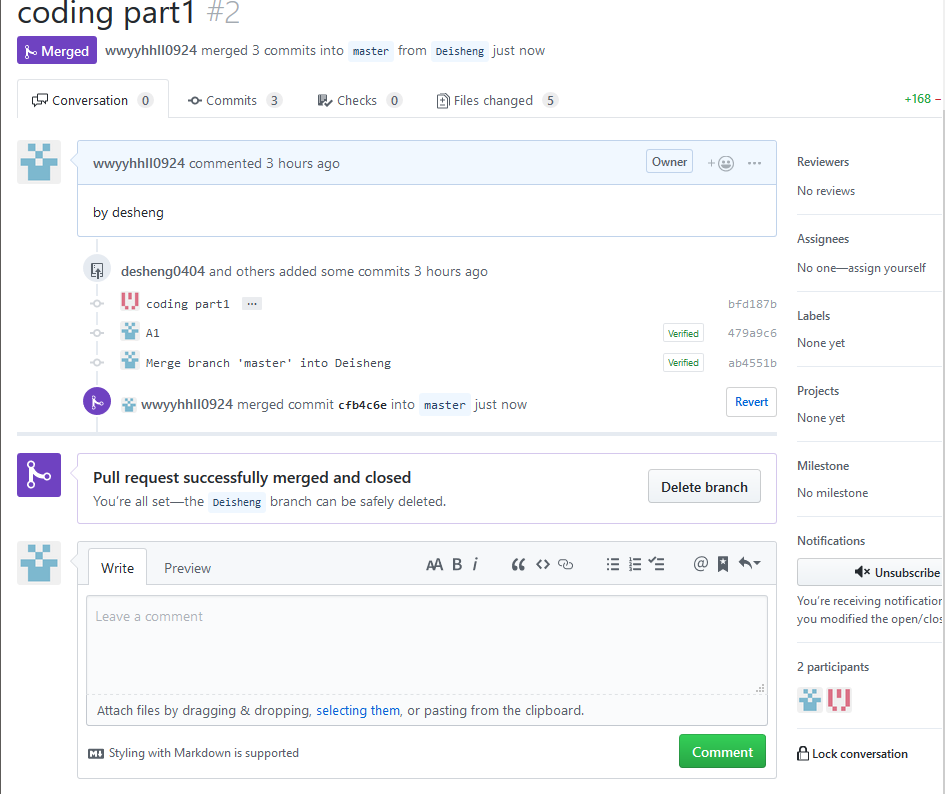
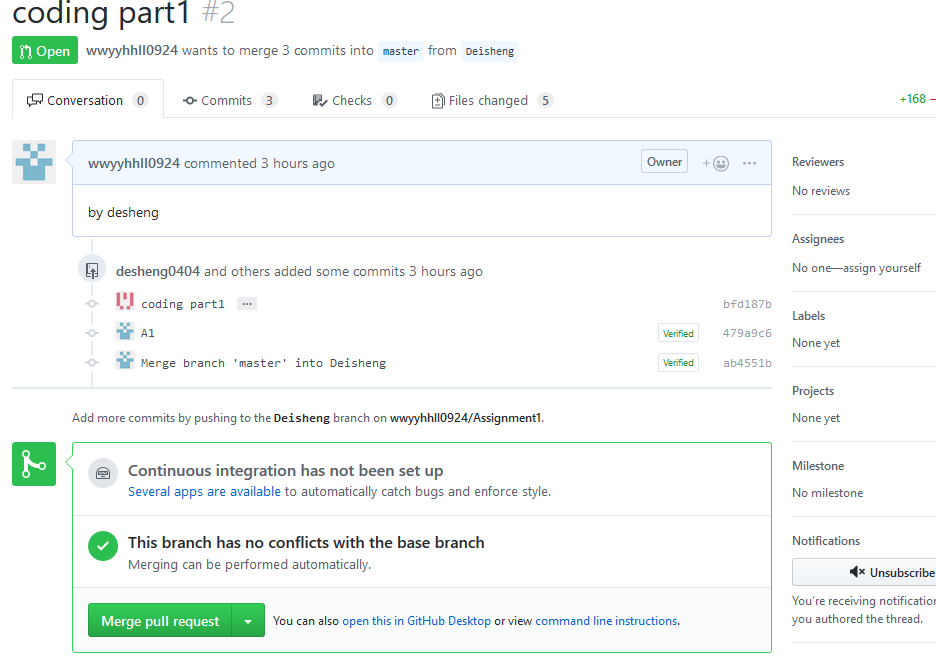
## Cloning repositories

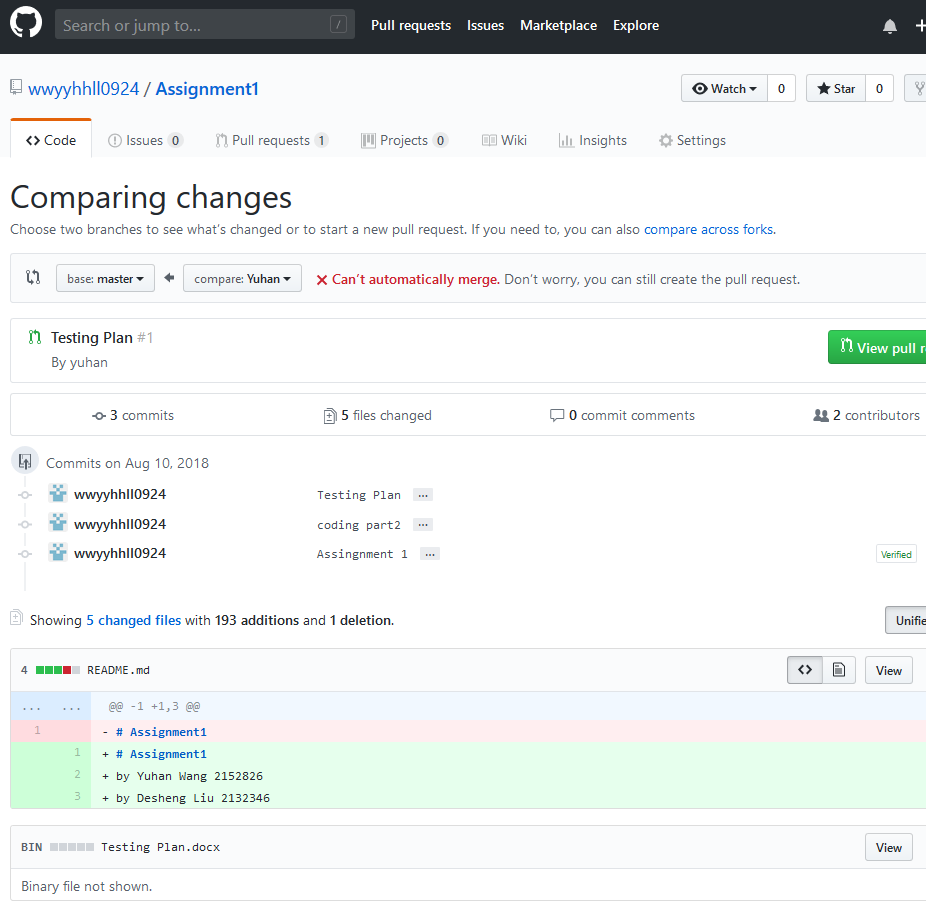


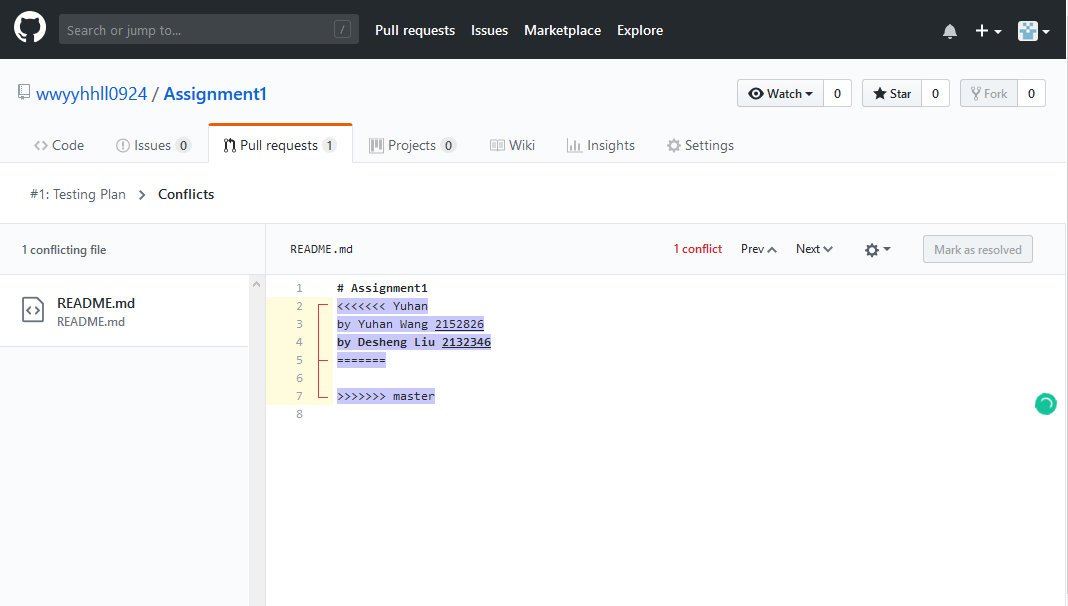


## Merge

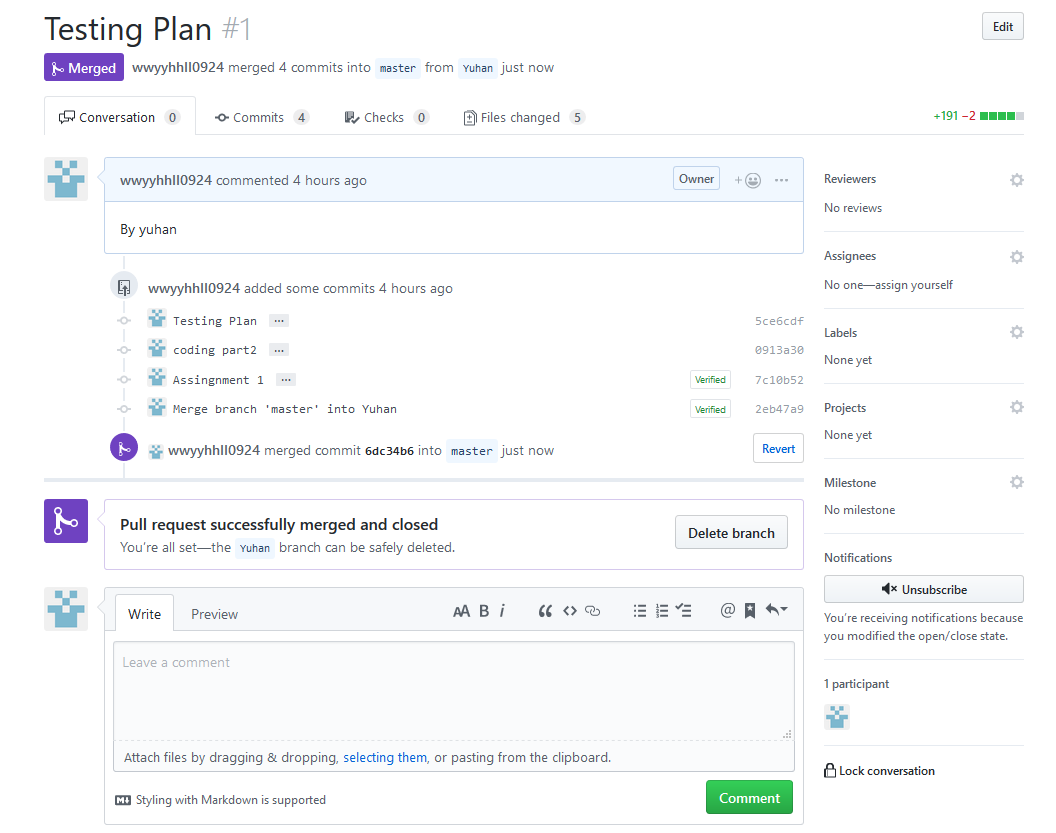


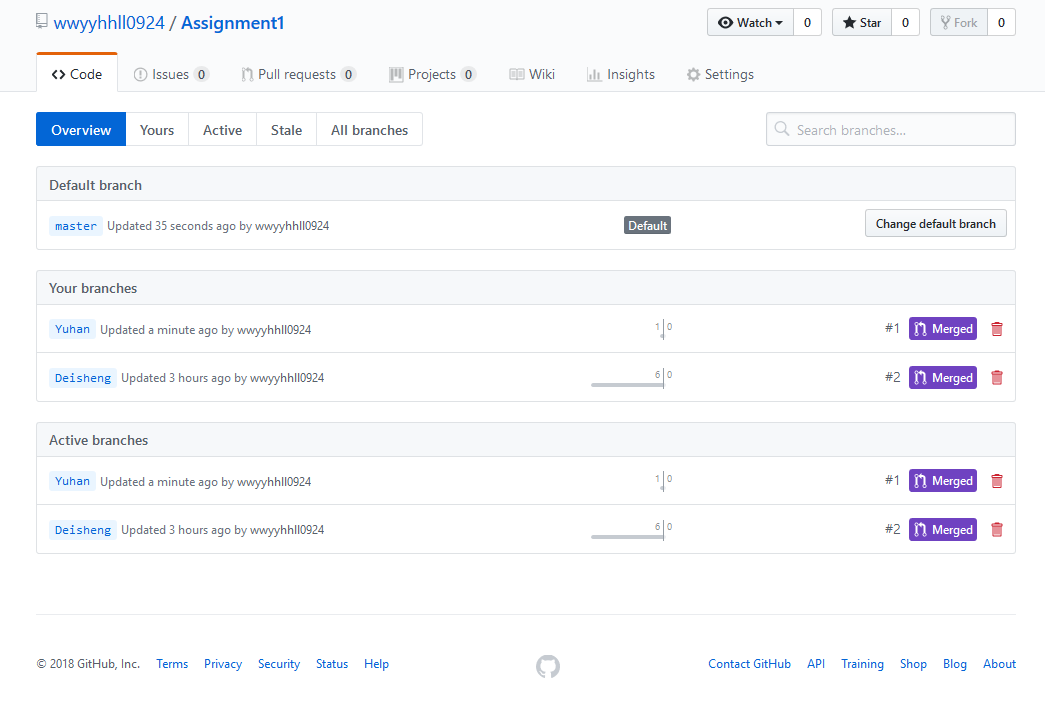






Delete and change the different contexts between master branch and yuhan branch for sync.





# Conclusion

During the first development phase, we learnt the team management skills and how to use time reasonable because my teammate also needs to work on the project, so we need to improve our efficiency. We used the simple progress report and the time sheet to track the work for every day, every two or three days we will organize the team meeting to talk about the current constraints we have, and help each other to solve the obstacles. At the end of the meeting we listed the action items for the next meeting and we will check the work in next meeting. Those necessary management skills we already learnt before, but during this development, we improved them. The most important thing is our team without any conflicts, everyone can work on their part responsible. A good relationship and community with the team member are essential.

For the development and technique parts, we still need to improve in the future. In current development, we did a good idea, and excellent design in the wireframe, however, is not anything on the wireframe has been implemented. I think the first reason is we know the theory of the knowledge but we cannot use the knowledge in the practical part efficiently, sometimes we need to research and spend lots of time on that. The second reason is we should not do too much assume next time. Sometimes we keep considering everything. If we can fix those two problems, we can do better in the future. In the technique part, after the project we can use Git now, this is a useful tool for sharing and tracking the coding work online.

We did testing when we finished the coding work. In the testing part, I spend some time to check the specification and requirements. Actually, lots of bugs will hide in the specification and requirements. If we cannot find the error the error will become fault then will become the failure to crash our system. I am using the Risk Matrix to demonstrate the risk in multi-dimension with three levels (high, mid, low) then depending on the risk level we can decide which part we need to test first and which part need most to test and which part does not need the test.

In the future implementation of the system, we will use the framework to make the UI (user interface) better and implement the navigation bar on every page. Planning to add more functions for checking attendance, checking the tutor timetable and booking an appointment with the tutor. Next phase also will add about, help and contact us pages in the navigation bar. For back-end, improve security and also need to think about the SQL injection problem. Those things will be implemented in the next phase. After that we think to create a small app for the system on Android platform.

Finally this is a memorable development experience for us, from this project we find which part we did well and which part we need to improve in the future.

# Appendix A

## Source code:

## Register.php

<?php include('server.php') ?>

<!DOCTYPE html>

<html>

<head>

<title>Registration system PHP and MySQL</title>

<link rel="stylesheet" type="text/css" href="style.css">

</head>

<body>

<div class="header">

<h2>Register</h2>

</div>

<form method="post" action="register.php">

<?php include('errors.php'); ?>

<div class="input-group">

<label>Username</label>

<input type="text" name="username" value="<?php echo $username; ?>">

</div>

<div class="input-group">

<label>Email</label>

<input type="email" name="email" value="<?php echo $email; ?>">

</div>

<div class="input-group">

<label>Password</label>

<input type="password" name="password\_1">

</div>

<div class="input-group">

<label>Confirm password</label>

<input type="password" name="password\_2">

</div>

<div class="input-group">

<button type="submit" class="btn" name="reg\_user">Register</button>

</div>

<p>

Already a member? <a href="login.php">Sign in</a>

</p>

</form>

</body>

</html>

## login.php:

<?php include('server.php') ?>

<!DOCTYPE html>

<html>

<head>

<title>Registration system PHP and MySQL</title>

<link rel="stylesheet" type="text/css" href="style.css">

</head>

<body>

<div class="header">

<h2>Login</h2>

</div>

<form method="post" action="login.php">

<?php include('errors.php'); ?>

<div class="input-group">

<label>Username</label>

<input type="text" name="username" >

</div>

<div class="input-group">

<label>Password</label>

<input type="password" name="password">

</div>

<div class="input-group">

<button type="submit" class="btn" name="login\_user">Login</button>

</div>

<p>

Not yet a member? <a href="register.php">Sign up</a>

</p>

</form>

</body>

</html>

## Index.php:

<?php

session\_start();

if (!isset($\_SESSION['username'])) {

$\_SESSION['msg'] = "You must log in first";

header('location: login.php');

}

if (isset($\_GET['logout'])) {

session\_destroy();

unset($\_SESSION['username']);

header("location: login.php");

}

?>

<!DOCTYPE html>

<html>

<head>

<title>Home</title>

<link rel="stylesheet" type="text/css" href="style.css">

</head>

<body>

<div class="header">

<h2>Result</h2>

</div>

<div class="content">

<?php if (isset($\_SESSION['success'])): ?>

<div class="error success">

<h3>

<?php

echo $\_SESSION['success'];

unset($\_SESSION['success'])

?>

</h3>

</div>

<?php endif ?>

<?php if (isset($\_SESSION['username'])): ?>

<div class="large-3 columns" style="text-align:right;">

<p><a href="index.php?logout='1'" style="color: red;" >Logout</a></p>

</div>

<div class="large-9 columns">

<p>Welcome <strong><?php echo $\_SESSION['username']; ?></strong></p><br>

<p>Welcome to Student result system! You can view your provisional results for courses in your current study and your past results.<br>

<br>

<br>

<p>&nbsp;0 - 49 D</p><p>50 - 59 C</p><p>60 - 64 C+</p><p>65 - 74 B</p><p>75 - 79 B+</p><p>80 - 89 A</p><p>90 - 100 A+</p>

<div class="large-12 columns">

<h2>Your Courses</h2>

</div>

<br>

<br>

<p>Software Development:&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;79%&nbsp;&nbsp;&nbsp;&nbsp;B+</P>

<p>Programming 3:&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;84%&nbsp;&nbsp;&nbsp;&nbsp;A</p>

<p>Advance Database:&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;68%&nbsp;&nbsp;&nbsp;&nbsp;B</p>

<p>Project Management:&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;

&nbsp;&nbsp;87%&nbsp;&nbsp;&nbsp;&nbsp;A</P>

<?php endif?>

</div>

</body>

</html>

## Server.php:

<?php

session\_start();

// initializing variables

$username = "";

$email = "";

$errors = array();

// connect to the database

$db = mysqli\_connect('localhost', 'root', '', 'registration');

// REGISTER USER

if (isset($\_POST['reg\_user'])) {

// receive all input values from the form

$username = mysqli\_real\_escape\_string($db, $\_POST['username']);

$email = mysqli\_real\_escape\_string($db, $\_POST['email']);

$password\_1 = mysqli\_real\_escape\_string($db, $\_POST['password\_1']);

$password\_2 = mysqli\_real\_escape\_string($db, $\_POST['password\_2']);

// form validation: ensure that the form is correctly filled ...

// by adding (array\_push()) corresponding error unto $errors array

if (empty($username)) { array\_push($errors, "Username is required"); }

if (empty($email)) { array\_push($errors, "Email is required"); }

if (empty($password\_1)) { array\_push($errors, "Password is required"); }

if ($password\_1 != $password\_2) {

array\_push($errors, "The two passwords do not match");

}

// first check the database to make sure

// a user does not already exist with the same username and/or email

$user\_check\_query = "SELECT \* FROM users WHERE username='$username' OR email='$email' LIMIT 1";

$result = mysqli\_query($db, $user\_check\_query);

$user = mysqli\_fetch\_assoc($result);

if ($user) { // if user exists

if ($user['username'] === $username) {

array\_push($errors, "Username already exists");

}

if ($user['email'] === $email) {

array\_push($errors, "email already exists");

}

}

// Finally, register user if there are no errors in the form

if (count($errors) == 0) {

$password = md5($password\_1);//encrypt the password before saving in the database

$query = "INSERT INTO users (username, email, password)

VALUES('$username', '$email', '$password')";

mysqli\_query($db, $query);

$\_SESSION['username'] = $username;

$\_SESSION['success'] = "You are now logged in";

header('location: index.php');

}

}

// LOGIN USER

if (isset($\_POST['login\_user'])) {

$username = mysqli\_real\_escape\_string($db, $\_POST['username']);

$password = mysqli\_real\_escape\_string($db, $\_POST['password']);

if (empty($username)) {

array\_push($errors, "Username is required");

}

if (empty($password)) {

array\_push($errors, "Password is required");

}

if (count($errors) == 0) {

$password = md5($password);

$query = "SELECT \* FROM users WHERE username='$username' AND password='$password'";

$results = mysqli\_query($db, $query);

if (mysqli\_num\_rows($results) == 1) {

$\_SESSION['username'] = $username;

header('location: index.php');

}else {

array\_push($errors, "Wrong username/password combination");

}

}

}

?>

## Style.css:

\* {

margin: 0px;

padding: 0px;

}

body {

font-size: 120%;

background: #F8F8FF;

}

.header {

width: 30%;

margin: 50px auto 0px;

color: white;

background: #5F9EA0;

text-align: center;

border: 1px solid #B0C4DE;

border-bottom: none;

border-radius: 10px 10px 0px 0px;

padding: 20px;

}

form, .content {

width: 30%;

margin: 0px auto;

padding: 20px;

border: 1px solid #B0C4DE;

background: white;

border-radius: 0px 0px 10px 10px;

}

.input-group {

margin: 10px 0px 10px 0px;

}

.input-group label {

display: block;

text-align: left;

margin: 3px;

}

.input-group input {

height: 30px;

width: 93%;

padding: 5px 10px;

font-size: 16px;

border-radius: 5px;

border: 1px solid gray;

}

.btn {

padding: 10px;

font-size: 15px;

color: white;

background: #5F9EA0;

border: none;

border-radius: 5px;

}

.error {

width: 92%;

margin: 0px auto;

padding: 10px;

border: 1px solid #a94442;

color: #a94442;

background: #f2dede;

border-radius: 5px;

text-align: left;

}

.success {

color: #3c763d;

background: #dff0d8;

border: 1px solid #3c763d;

margin-bottom: 20px;

}

## errors.php:

<?php if (count($errors) > 0) : ?>

<div class="error">

<?php foreach ($errors as $error) : ?>

<p><?php echo $error ?></p>

<?php endforeach ?>

</div>

<?php endif ?>

## users.sql:

-- phpMyAdmin SQL Dump

-- version 4.1.14

-- http://www.phpmyadmin.net

--

-- Host: 127.0.0.1

-- Generation Time: Aug 08, 2018 at 05:27 AM

-- Server version: 5.6.17

-- PHP Version: 5.5.12

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8 \*/;

--

-- Database: `registration`

--

-- --------------------------------------------------------

--

-- Table structure for table `users`

--

CREATE TABLE IF NOT EXISTS `users` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`username` varchar(255) NOT NULL,

`email` varchar(255) NOT NULL,

`password` varchar(255) NOT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=10 ;

--

-- Dumping data for table `users`

--

INSERT INTO `users` (`id`, `username`, `email`, `password`) VALUES

(1, 'desheng', 'desheng@example.com', '4682cab6df5eefb3974201ac60830157'),

(9, 'yuhanwang', 'yuhan@gmail.com', '71fc54759b978f924aa8e2d5e2cef0d3');

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

# Appendix B

**GitHub Links:**

**Desheng Liu:** [**https://github.com/desheng0404**](https://github.com/desheng0404)

**Yuhan Wang:** [**https://github.com/wwyyhhll0924**](https://github.com/wwyyhhll0924)