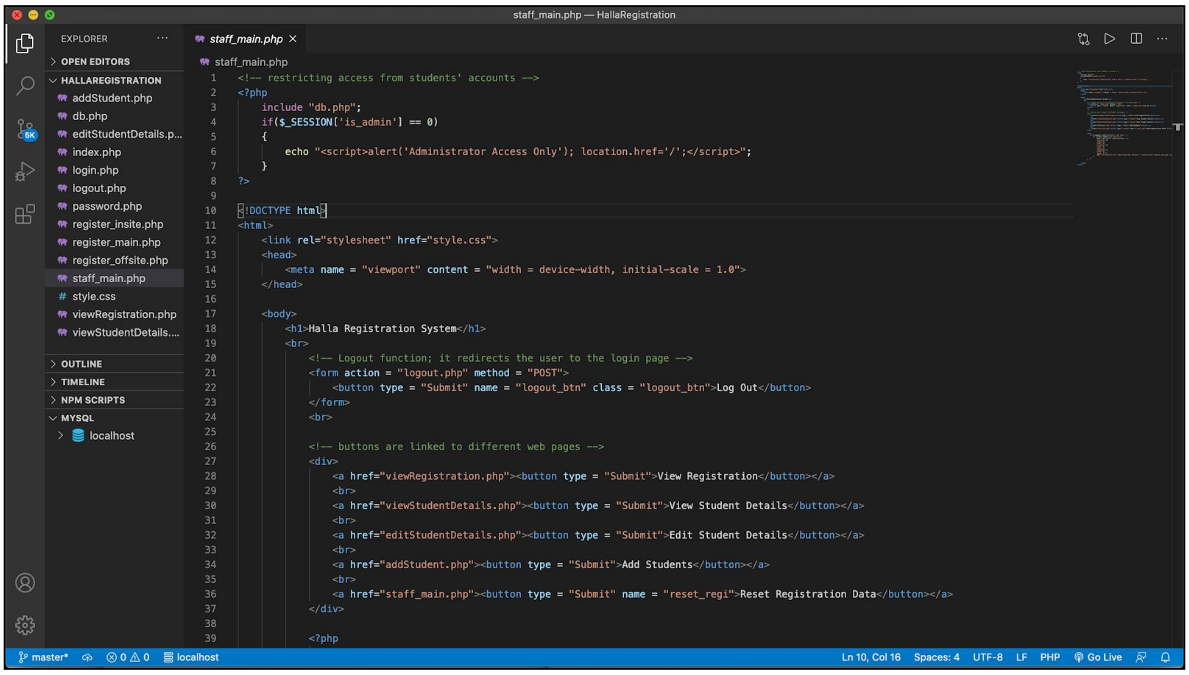
List of techniques used in the project

- Increasing usability with the use of GUI
- Restricting access from student accounts - security
- Algorithmic Sorting to list student data
- Checking user action with an alert
- Open-Source Encryption algorithm for security
- Use of <meta> for compatibility with mobile devices
- Functions for abstraction and code reusability

External services used in the project

During the development of the program, a code editor 'Visual Studio Code' is used since it supports HTML, CSS and PHP at once.



```
<!-- restricting access from students' accounts -->
<?php
    include "db.php";
    if($_SESSION['is_admin'] == 0)
    {
        echo "<script>alert('Administrator Access Only'); location.href='/';</script>";
    }
?>

<!DOCTYPE html>
<html>
    <link rel="stylesheet" href="style.css">
    <head>
        <meta name = "viewport" content = "width = device-width, initial-scale = 1.0">
    </head>

    <body>
        <h1>Halla Registration System</h1>
        <br>
            <!-- Logout function; it redirects the user to the login page -->
            <form action = "logout.php" method = "POST">
                <button type = "Submit" name = "logout_btn" class = "logout_btn">Log Out</button>
            </form>
            <br>

            <!-- buttons are linked to different web pages -->
            <div>
                <a href="viewRegistration.php"><button type = "Submit">View Registration</button></a>
                <br>
                <a href="viewStudentDetails.php"><button type = "Submit">View Student Details</button></a>
                <br>
                <a href="editStudentDetails.php"><button type = "Submit">Edit Student Details</button></a>
                <br>
                <a href="addStudent.php"><button type = "Submit">Add Students</button></a>
                <br>
                <a href="staff_main.php"><button type = "Submit" name = "reset_regi">Reset Registration Data</button></a>
            </div>

            <?php
```

Figure 1.1

'phpMyAdmin' is used when administering databases in MySQL for web hosting service. It offers various features such as browsing, creating, dropping and editing databases with an intuitive web interface.

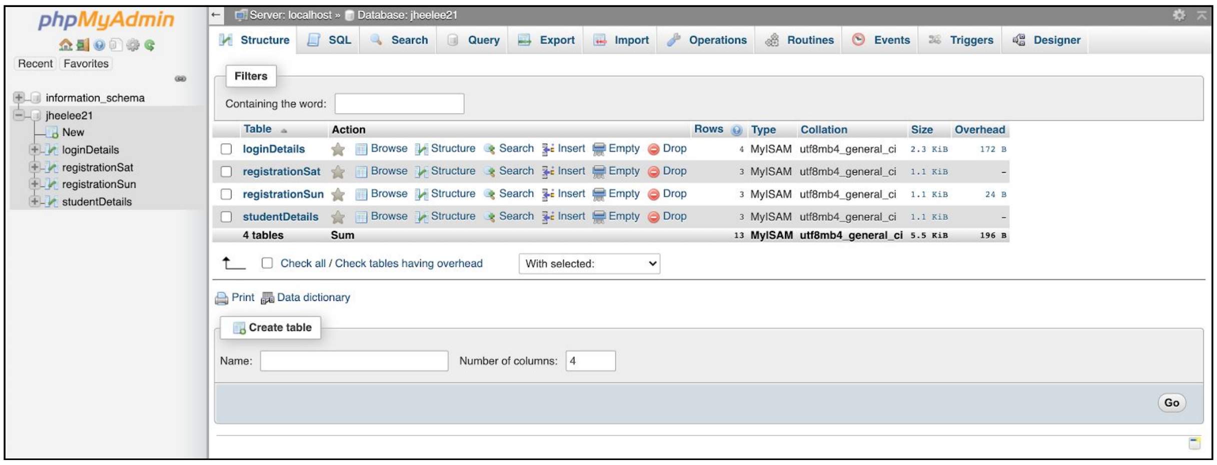
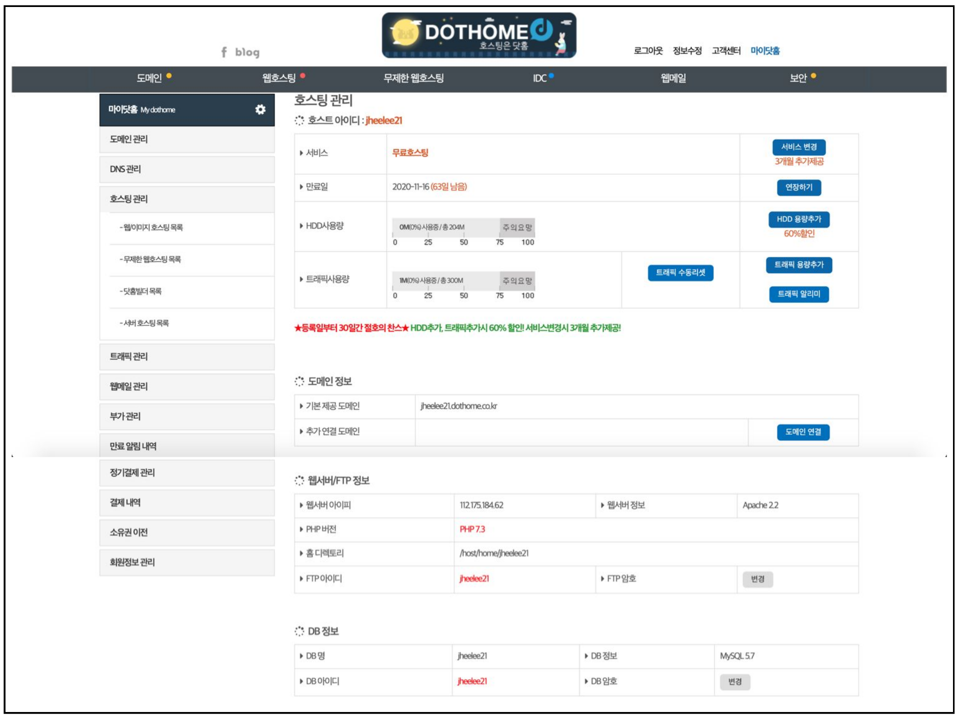

Server: localhost » Database: jheelee21

	Table	Action	Rows	Type	Collation	Size	Overhead
☐	loginDetails	Browse Structure Search Insert Empty Drop	4	MyISAM	utf8mb4_general_ci	2.3 KiB	172 B
☐	registrationSat	Browse Structure Search Insert Empty Drop	3	MyISAM	utf8mb4_general_ci	1.1 KiB	-
☐	registrationSun	Browse Structure Search Insert Empty Drop	3	MyISAM	utf8mb4_general_ci	1.1 KiB	24 B
☐	studentDetails	Browse Structure Search Insert Empty Drop	3	MyISAM	utf8mb4_general_ci	1.1 KiB	-
	4 tables	Sum	13	MyISAM	utf8mb4_general_ci	5.5 KiB	196 B

Create table — Name: Number of columns: 4

Figure 1.2

‘Dothome’ is a free web hosting service, supporting about 30 users at the same time. There won’t be any disadvantage caused by small traffic size since the users for my product falls below 30(2 staff, 25 students).


호스팅 관리

호스트 아이디 : jheelee21

서비스	무료호스팅	서비스 변경 3개월 추가제공
만료일	2020-11-16 (63일 남음)	연장하기
HDD사용량	0M(0%) 사용중 / 총 204M	HDD 용량추가 60%할인
트래픽사용량	1M(0%) 사용중 / 총 300M	트래픽 용량추가 트래픽 알리미

★등록일부터 30일간 절호의 찬스★ HDD추가, 트래픽추가시 60% 할인! 서비스변경시 3개월 추가제공!

도메인 정보

기본 제공 도메인	jheelee21.dothome.co.kr
추가 연결 도메인	

웹서버/FTP 정보

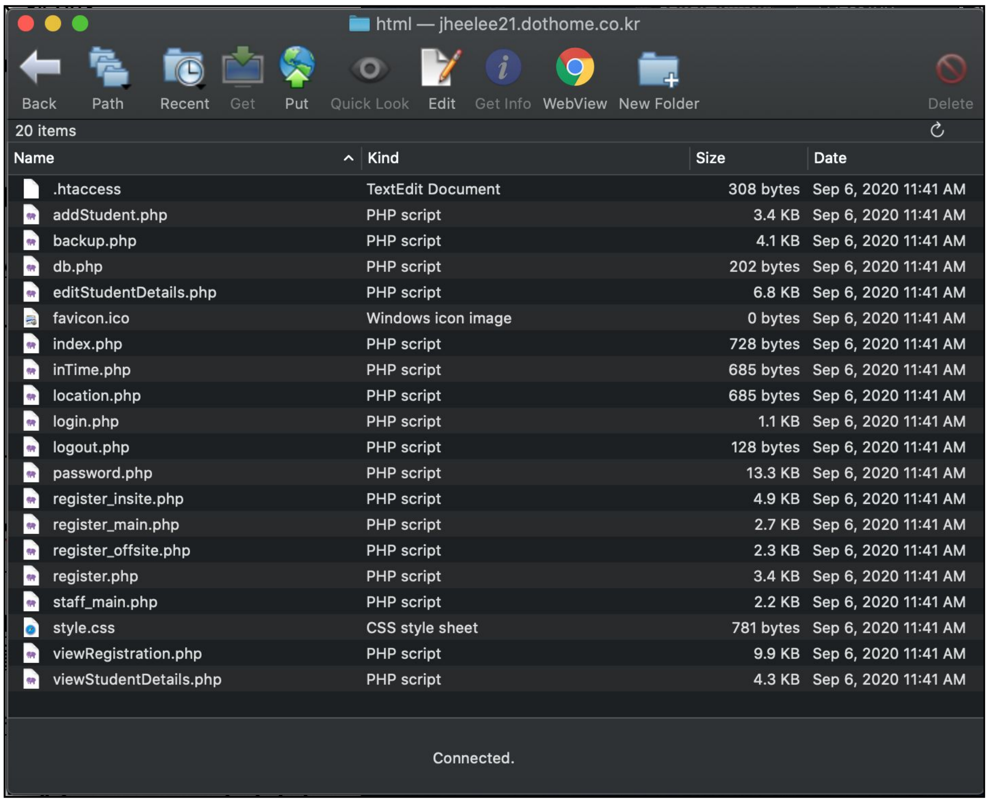
웹서버 아이피	112.175.184.62	웹서버 정보	Apache 2.2
PHP 버전	PHP 7.3		
홈 디렉토리	/host/home/jheelee21		
FTP 아이디	jheelee21	FTP 암호	변경

DB 정보

DB 명	jheelee21	DB 정보	MySQL 5.7
DB 아이디	jheelee21	DB 암호	변경

Figure 1.3

‘Fetch’ is a free File Transfer Protocol client that is used to transfer files between the client and the server. It is used to upload the code from the local server to the webserver provided by ‘Dothome’.



20 items

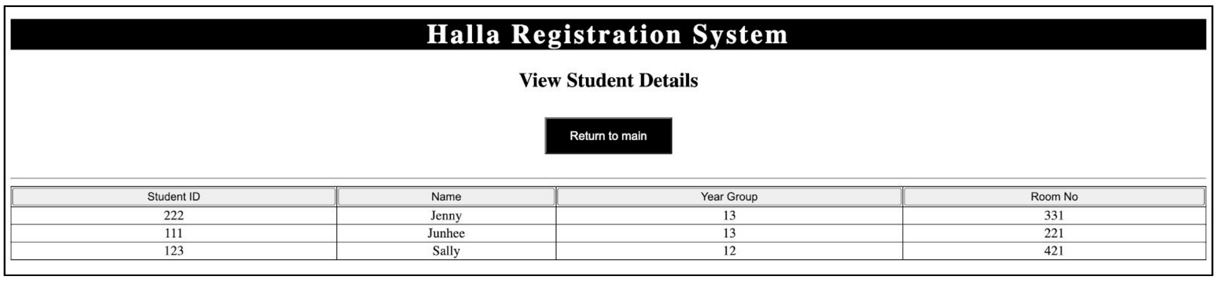
Name	Kind	Size	Date
.htaccess	TextEdit Document	308 bytes	Sep 6, 2020 11:41 AM
addStudent.php	PHP script	3.4 KB	Sep 6, 2020 11:41 AM
backup.php	PHP script	4.1 KB	Sep 6, 2020 11:41 AM
db.php	PHP script	202 bytes	Sep 6, 2020 11:41 AM
editStudentDetails.php	PHP script	6.8 KB	Sep 6, 2020 11:41 AM
favicon.ico	Windows icon image	0 bytes	Sep 6, 2020 11:41 AM
index.php	PHP script	728 bytes	Sep 6, 2020 11:41 AM
inTime.php	PHP script	685 bytes	Sep 6, 2020 11:41 AM
location.php	PHP script	685 bytes	Sep 6, 2020 11:41 AM
login.php	PHP script	1.1 KB	Sep 6, 2020 11:41 AM
logout.php	PHP script	128 bytes	Sep 6, 2020 11:41 AM
password.php	PHP script	13.3 KB	Sep 6, 2020 11:41 AM
register_insite.php	PHP script	4.9 KB	Sep 6, 2020 11:41 AM
register_main.php	PHP script	2.7 KB	Sep 6, 2020 11:41 AM
register_offsite.php	PHP script	2.3 KB	Sep 6, 2020 11:41 AM
register.php	PHP script	3.4 KB	Sep 6, 2020 11:41 AM
staff_main.php	PHP script	2.2 KB	Sep 6, 2020 11:41 AM
style.css	CSS style sheet	781 bytes	Sep 6, 2020 11:41 AM
viewRegistration.php	PHP script	9.9 KB	Sep 6, 2020 11:41 AM
viewStudentDetails.php	PHP script	4.3 KB	Sep 6, 2020 11:41 AM

Connected.

Figure 1.4

Increasing usability with the use of GUI

To create a user-friendly interface, GUI such as buttons and tables are used in the product.



Halla Registration System

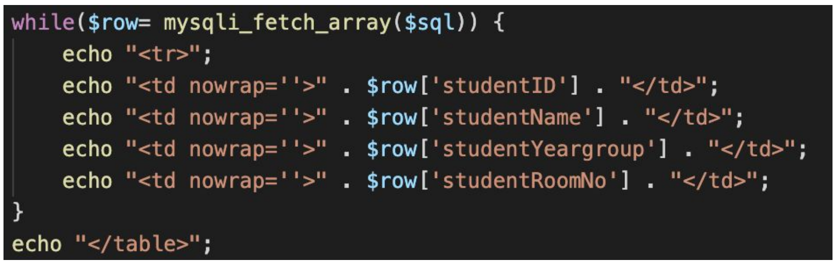
View Student Details

Return to main

Student ID	Name	Year Group	Room No
222	Jenny	13	331
111	Junhee	13	221
123	Sally	12	421

Figure 2.1

Tables are used to display a lot of information at once, such as the details for all the students stored in the database. This allows the user to understand the data in a glance since the columns are divided up into specific categories. Figure 2.2 is the code used to create a table of the details from the database server.



```
while($row= mysqli_fetch_array($sql)) {
    echo "<tr>";
    echo "<td nowrap=''>" . $row['studentID'] . "</td>";
    echo "<td nowrap=''>" . $row['studentName'] . "</td>";
    echo "<td nowrap=''>" . $row['studentYeargroup'] . "</td>";
    echo "<td nowrap=''>" . $row['studentRoomNo'] . "</td>";
}
echo "</table>";
```

Figure 2.2

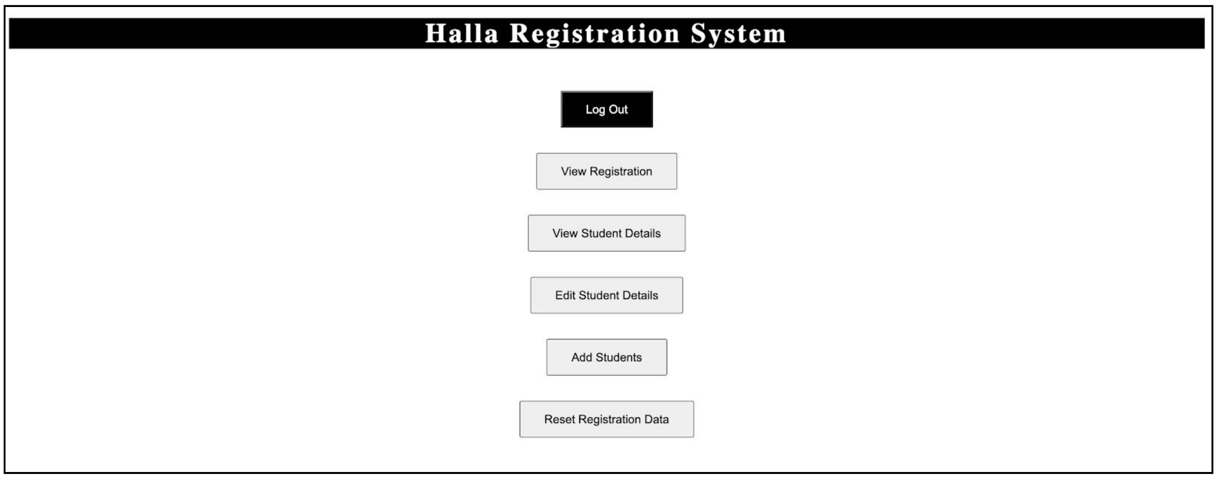
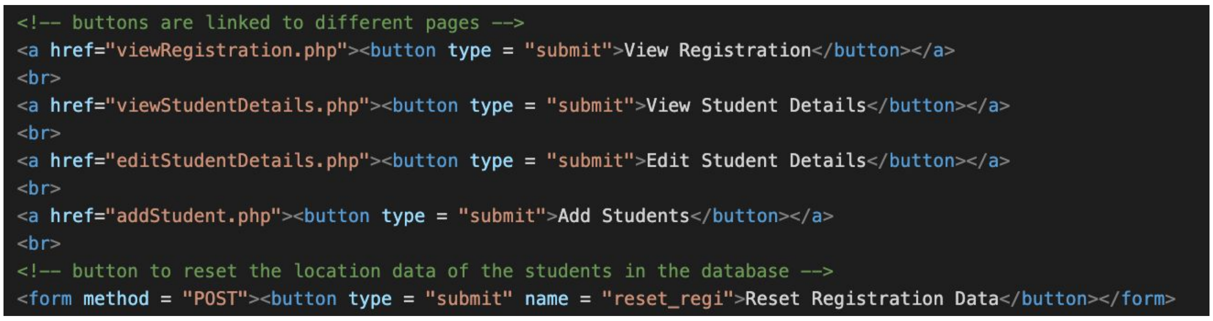


Figure 2.3

Furthermore, every action that the user can perform is enabled by labelled buttons, for example, 'View Student Details' button in Figure 2.2. Also, All the buttons are labelled, directing the user to select the desired action. Just clicking on the button triggers functions of the program, hence making it easier for the user to utilize the program. The code for the buttons is shown in Figure 2.4.



```html
<!-- buttons are linked to different pages -->
<a href="viewRegistration.php"><button type = "submit">View Registration</button></a>
<br>
<a href="viewStudentDetails.php"><button type = "submit">View Student Details</button></a>
<br>
<a href="editStudentDetails.php"><button type = "submit">Edit Student Details</button></a>
<br>
<a href="addStudent.php"><button type = "submit">Add Students</button></a>
<br>
<!-- button to reset the location data of the students in the database -->
<form method = "POST"><button type = "submit" name = "reset_regi">Reset Registration Data</button></form>
```

Figure 2.4

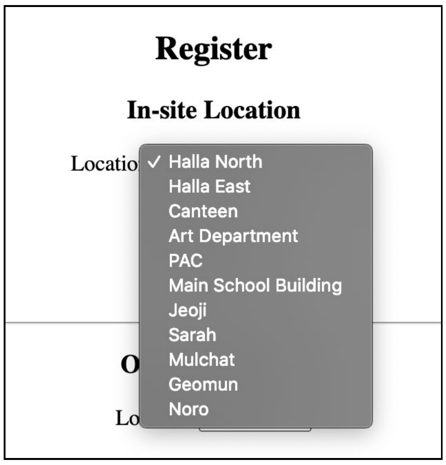
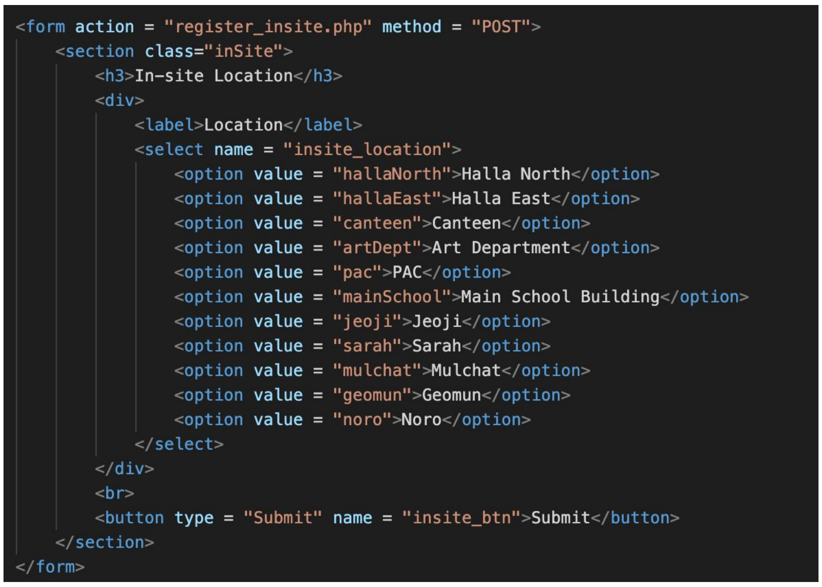


Figure 2.5

The drop-down box is used when the user is given the predetermined options to select their status from, such as locations and year groups. This prevents students from inputting an incorrect location or making a typing error.

<select> and <option> is used in the code to enable the use of this GUI. (Figure 2.6)

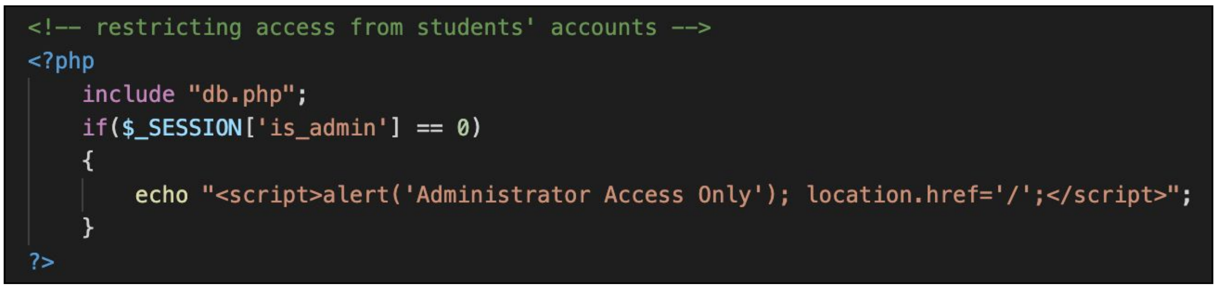


```html
<form action = "register_insite.php" method = "POST">
    <section class="inSite">
        <h3>In-site Location</h3>
        <div>
            <label>Location</label>
            <select name = "insite_location">
                <option value = "hallaNorth">Halla North</option>
                <option value = "hallaEast">Halla East</option>
                <option value = "canteen">Canteen</option>
                <option value = "artDept">Art Department</option>
                <option value = "pac">PAC</option>
                <option value = "mainSchool">Main School Building</option>
                <option value = "jeoji">Jeoji</option>
                <option value = "sarah">Sarah</option>
                <option value = "mulchat">Mulchat</option>
                <option value = "geomun">Geomun</option>
                <option value = "noro">Noro</option>
            </select>
        </div>
        <br>
        <button type = "Submit" name = "insite_btn">Submit</button>
    </section>
</form>
```

Figure 2.6

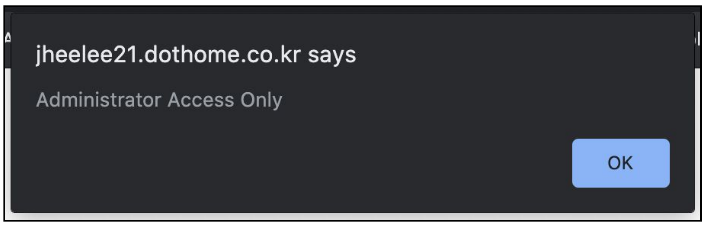
Restricting access from student accounts

The program supports viewing student details, viewing registration data and editing student details only to administrator accounts, which are given to the staff.



```
<!-- restricting access from students' accounts -->
<?php
    include "db.php";
    if($_SESSION['is_admin'] == 0)
    {
        echo "<script>alert('Administrator Access Only'); location.href='/';</script>";
    }
?>
```

Figure 3.1



jheelee21.dothome.co.kr says

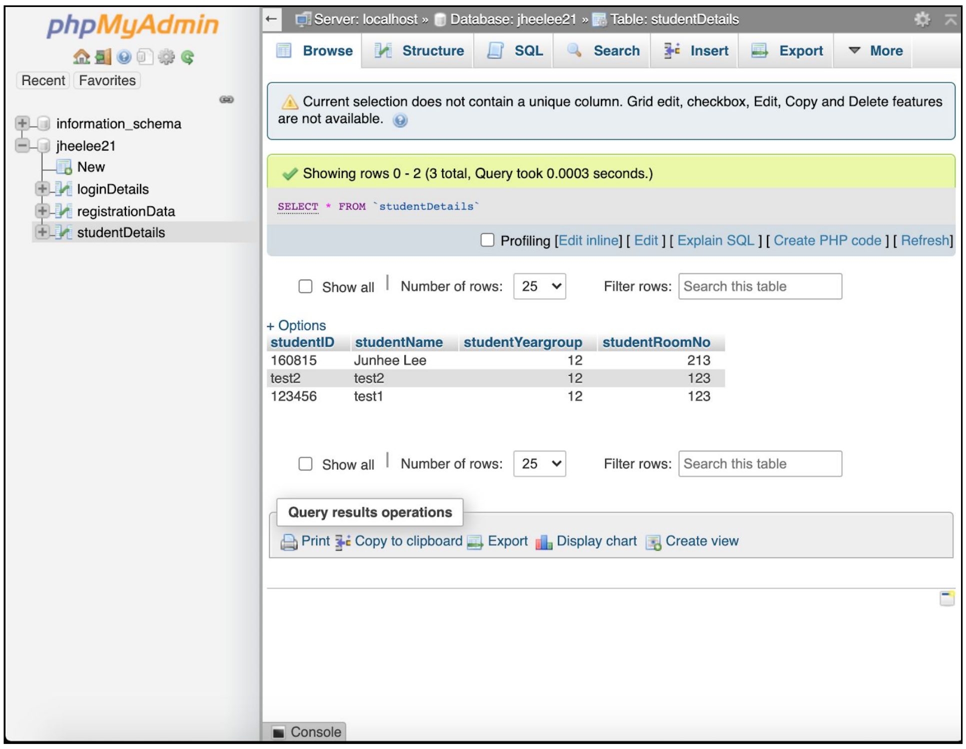
Administrator Access Only

Figure 3.2

This function guarantees authenticity when accessing, editing or adding student details and registration data. The access is restricted even when a student account tries to access the staff-only web pages through the domain name.

Algorithmic Sorting to list student data

The program displays the student details in a table, which is fetched from the data stored in the MySQL database in the PHPMyAdmin server.



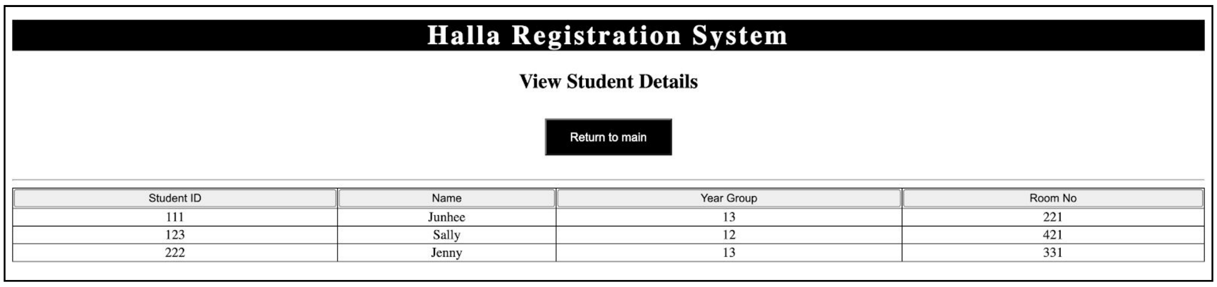
Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 2 (3 total, Query took 0.0003 seconds.)

```
SELECT * FROM `studentDetails`
```

studentID	studentName	studentYeargroup	studentRoomNo
160815	Junhee Lee	12	213
test2	test2	12	123
123456	test1	12	123

Figure 4.1

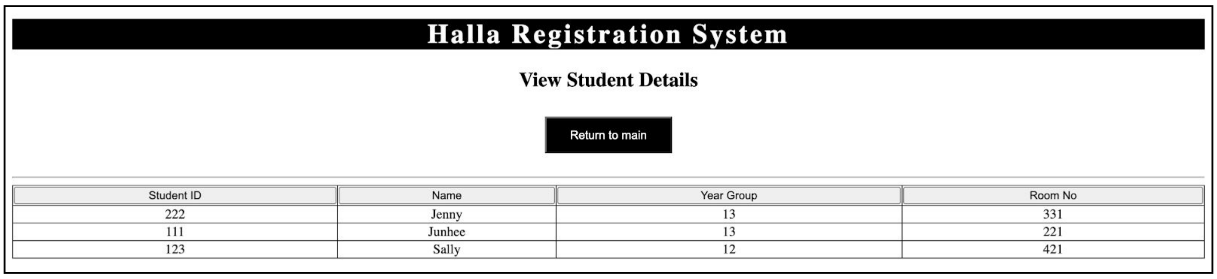


Student ID	Name	Year Group	Room No
111	Junhee	13	221
123	Sally	12	421
222	Jenny	13	331

Figure 4.2

When the user clicks the table header the data in the table is sorted in ascending order by the variable that the user chose. A table in Figure 4.2 is sorted by student ID.

Figure 4.3 is when the user clicked ‘Name’; the table is sorted by the alphabetical order of Student names.

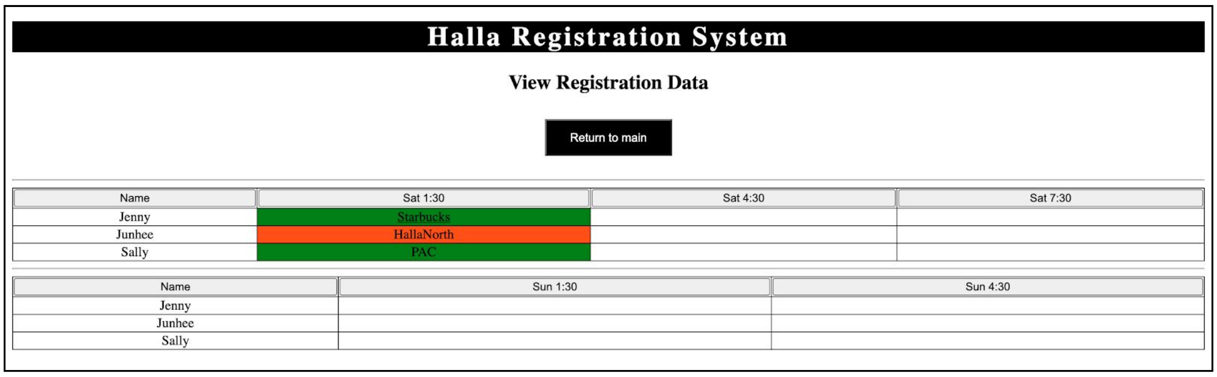

Halla Registration System

View Student Details

Student ID	Name	Year Group	Room No
222	Jenny	13	331
111	Junhee	13	221
123	Sally	12	421

Figure 4.3

This applies to registration data as well, as shown below.


Halla Registration System

View Registration Data

Name	Sat 1:30	Sat 4:30	Sat 7:30
Jenny	Starbucks		
Junhee	HallaNorth		
Sally	PAC		

Name	Sun 1:30	Sun 4:30
Jenny		
Junhee		
Sally		

Figure 4.4

The sorting function is done by ‘order by’ command in MySQL, it sorts the records in ascending order by default.

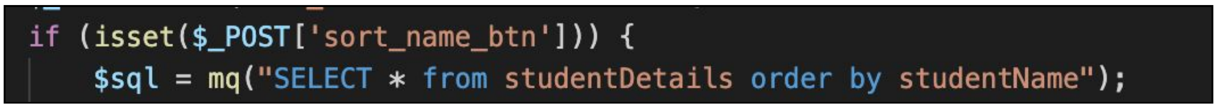
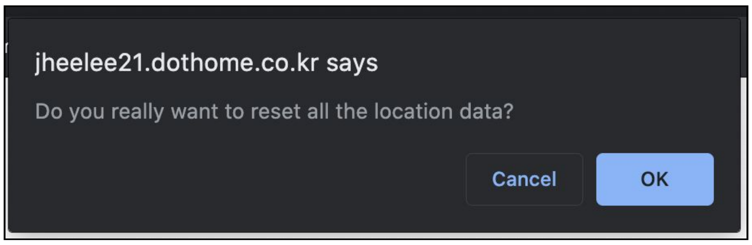

```
if (isset($_POST['sort_name_btn'])) {
    $sql = mq("SELECT * from studentDetails order by studentName");
```

Figure 4.5

By sorting function, users can view the data in a more organized way and they can group the data according to required actions such as finding students’ rooms on the second floor of the boarding house. This function allows the fourth success criteria to be accomplished; ‘The administrator can choose how to sort the table of data’.

Checking user action with an alert

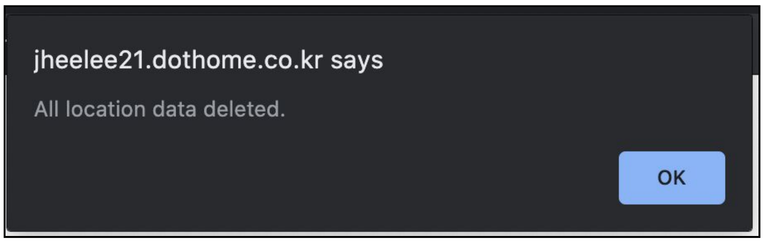
For ‘reset registration data’ and ‘reset all student data’ function, when the button is clicked, an alert pops up. As resetting the data in the database cannot be undone once deleted, the system needs an extra step to re-check the user's action.


jheelee21.dothome.co.kr says

Do you really want to reset all the location data?

Figure 5.1

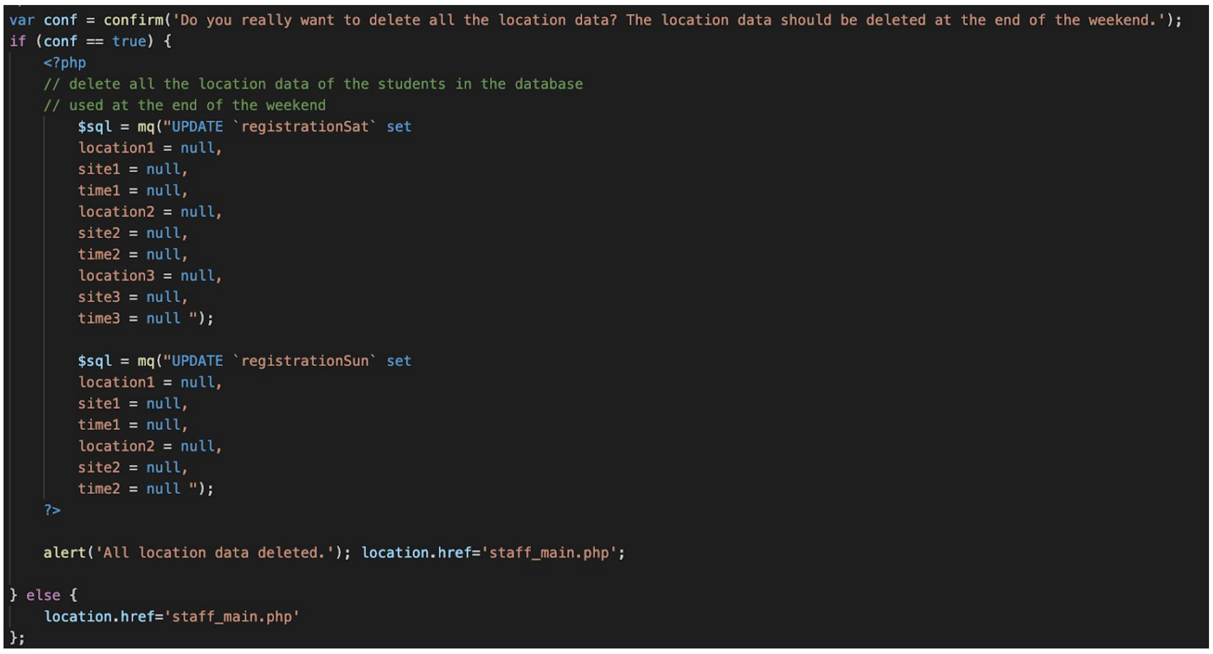
Double-checking the action prevents data loss from mistakes. The data is deleted only if the ‘OK’ button is clicked from the pop-up alert (Figure 5.2).


jheelee21.dothome.co.kr says

All location data deleted.

Figure 5.2

Using an embedded function in php, ‘confirm()’, boolean value for the user’s decision from the alert box is passed (Figure 5.3). Then, an if statement is utilized to check whether the location data should be deleted or not. ‘UPDATE’, which is a MySQL query, is used to manipulate the location data stored in the database server.



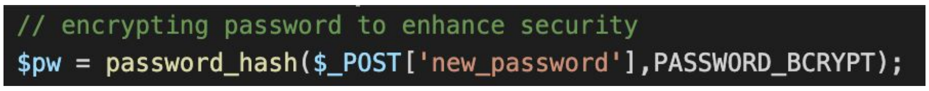
```
var conf = confirm('Do you really want to delete all the location data? The location data should be deleted at the end of the weekend.');
if (conf == true) {
    <?php
    // delete all the location data of the students in the database
    // used at the end of the weekend
        $sql = mq("UPDATE `registrationSat` set
        location1 = null,
        site1 = null,
        time1 = null,
        location2 = null,
        site2 = null,
        time2 = null,
        location3 = null,
        site3 = null,
        time3 = null ");

        $sql = mq("UPDATE `registrationSun` set
        location1 = null,
        site1 = null,
        time1 = null,
        location2 = null,
        site2 = null,
        time2 = null ");
    ?>

    alert('All location data deleted.'); location.href='staff_main.php';

} else {
    location.href='staff_main.php'
};
```

Figure 5.3

Open Source Encryption algorithm for security

The use of password encrypting enhances the security of my product, by preventing the leakage of user login details even when unauthorized individuals access the database of the program.

When the administrator account adds a new student to the database, the password of the student account should be entered. The input password is then encrypted by the code below.

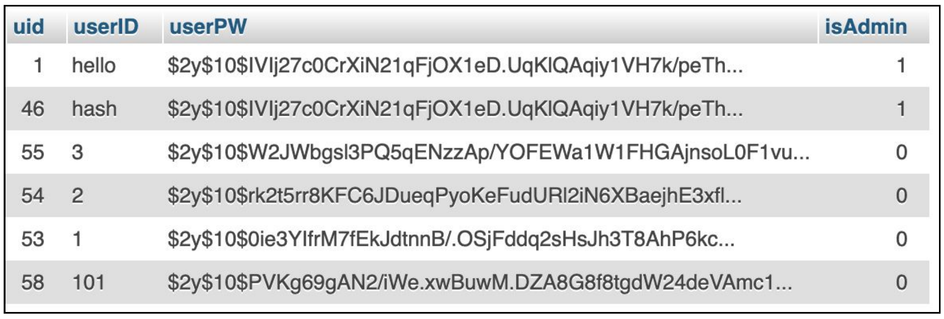


```
// encrypting password to enhance security
$pw = password_hash($_POST['new_password'],PASSWORD_BCRYPT);
```

Figure 6.1

'password.php' is an open-source code from Anthony Ferrara[1], designed to create password hashes. 'password_hash' is the function from 'password.php' and 'PASSWORD_BCRYPT' is the parameter used to encrypt the string in the first parameter.

The hashed password can be stored in the database (Figure 6.2) and in the login process, the input password can be hashed using the same method and compared with the stored data.



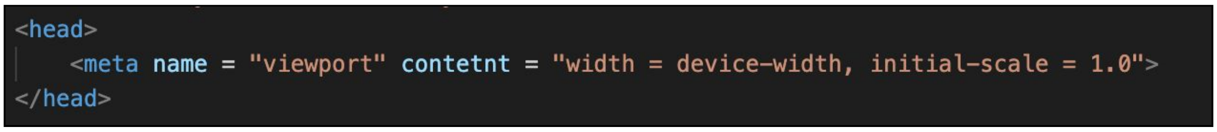
uid	userID	userPW	isAdmin
1	hello	$2y$10$IVIj27c0CrXiN21qFjOX1eD.UqKIQAqiy1VH7k/peTh...	1
46	hash	$2y$10$IVIj27c0CrXiN21qFjOX1eD.UqKIQAqiy1VH7k/peTh...	1
55	3	$2y$10$W2JWbgsl3PQ5qENzzAp/YOFEWa1W1FHGAjnsoL0F1vu...	0
54	2	$2y$10$rk2t5rr8KFC6JDueqPyoKeFudURl2iN6XBaejhE3xfl...	0
53	1	$2y$10$0ie3YIfrM7fEkJdtnnB/.OSjFddq2sHsJh3T8AhP6kc...	0
58	101	$2y$10$PVKg69gAN2/iWe.xwBuwM.DZA8G8f8tgdW24deVAmc1...	0

Figure 6.2

[1] GitHub. 2021. Ircmaxell/Password_Compat. [online] Available at: <https://github.com/ircmaxell/password_compat> [Accessed 20 November 2020].

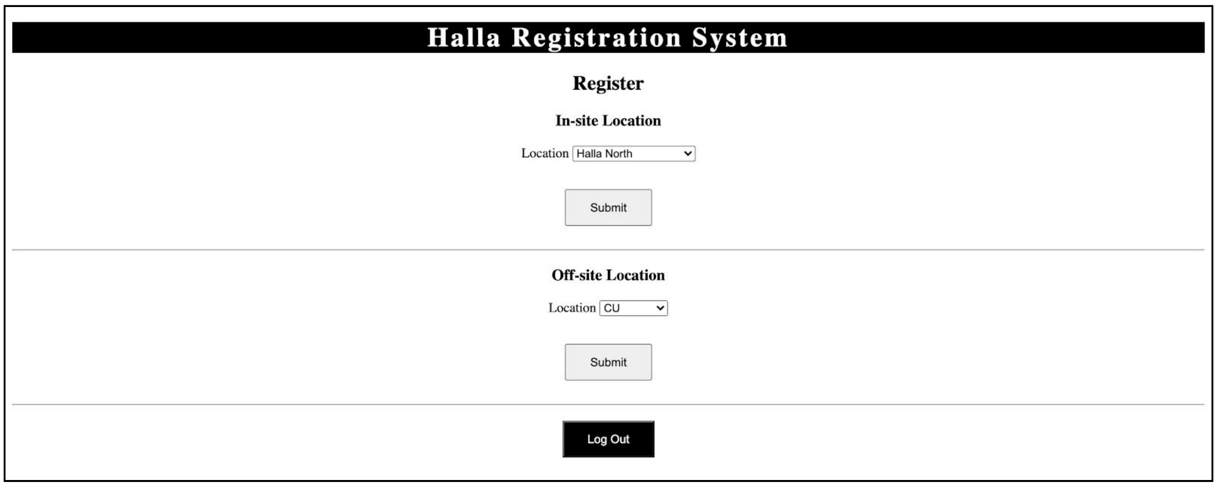
Use of <meta> for compatibility with mobile devices

In order to increase the usability of the program from a mobile phone, the display of every web page implemented a <meta> tag. It sets the viewport which is the user's visible area of a web page based on the screen size of the device and controls the scaling of the elements in the web page.



```
<head>
    <meta name = "viewport" contetnt = "width = device-width, initial-scale = 1.0">
</head>
```

Figure 7.1



Halla Registration System

Register

In-site Location

Location Halla North

Submit

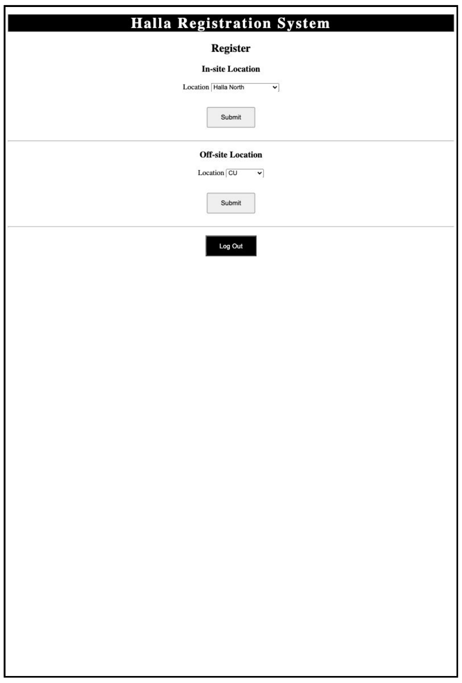
Off-site Location

Location CU

Submit

Log Out

Figure 7.2



Halla Registration System

Register

In-site Location

Location Halla North

Submit

Off-site Location

Location CU

Submit

Log Out

Before applying the <meta> tag to the program, the display on a laptop screen does not seem to be an issue during the usage, yet when viewed on a small-sized screen as if in mobile devices, texts and buttons are too small to click on. Hence, the user could experience difficulties in reading the texts or tapping on the desired element by the touch screen.

Figure 7.3

After applying the <meta> tag to ‘register_main.php’, the display of the web page is enlarged, thus it is better for the user to read the texts and tap on the elements to submit their location data.

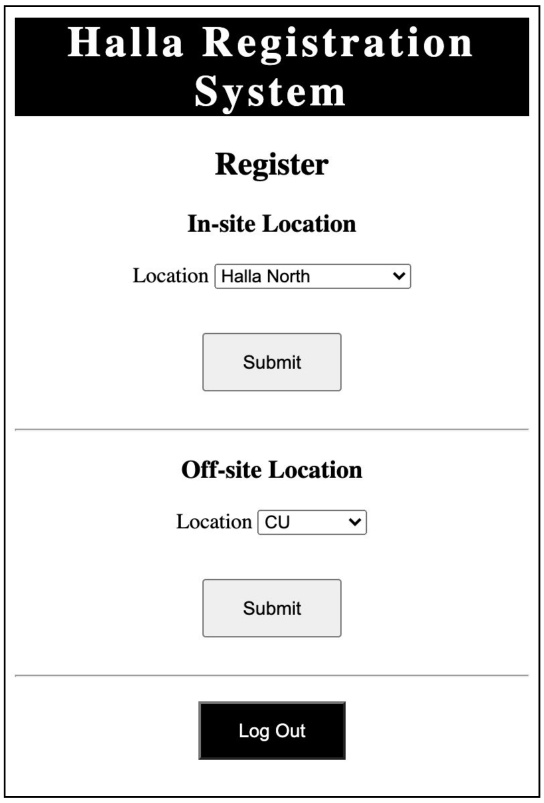
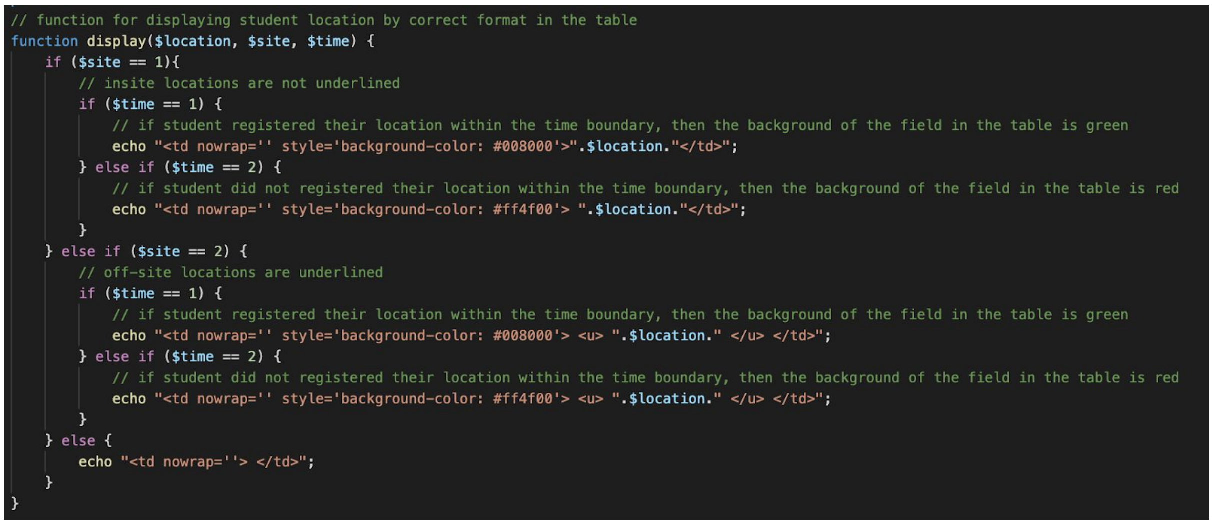


Figure 7.4

Functions for abstraction and code reusability

Creating functions increases the reusability of the code while making the programming process faster and easier by reducing the redundancy and complexity of the code.

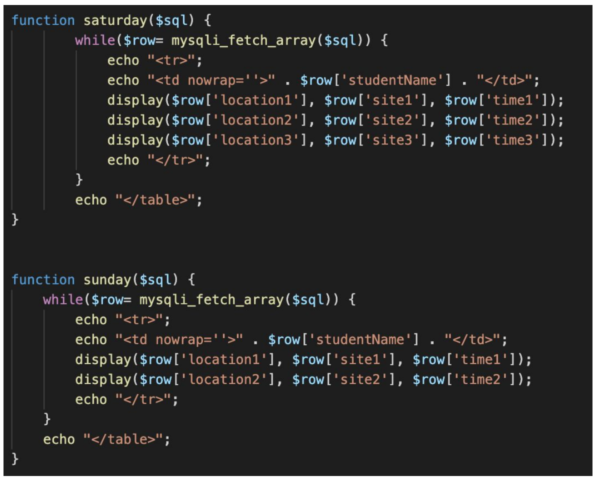
In my product, for displaying student location data on the table, function ‘display’ is used with the parameters of location, site and time. (Figure 8.1) For each row, 5 maximum locations are entered in a table with appropriate box colour and font according to the site and registered time. By using if conditions, he style of the box is decided in the function ‘display’ according to the parameters.



```php
// function for displaying student location by correct format in the table
function display($location, $site, $time) {
    if ($site == 1){
        // insite locations are not underlined
        if ($time == 1) {
            // if student registered their location within the time boundary, then the background of the field in the table is green
            echo "<td nowrap='' style='background-color: #008000'>".$location."</td>";
        } else if ($time == 2) {
            // if student did not registered their location within the time boundary, then the background of the field in the table is red
            echo "<td nowrap='' style='background-color: #ff4f00'> ".$location."</td>";
        }
    } else if ($site == 2) {
        // off-site locations are underlined
        if ($time == 1) {
            // if student registered their location within the time boundary, then the background of the field in the table is green
            echo "<td nowrap='' style='background-color: #008000'> <u> ".$location." </u> </td>";
        } else if ($time == 2) {
            // if student did not registered their location within the time boundary, then the background of the field in the table is red
            echo "<td nowrap='' style='background-color: #ff4f00'> <u> ".$location." </u> </td>";
        }
    } else {
        echo "<td nowrap=''> </td>";
    }
}
```

Figure 8.1

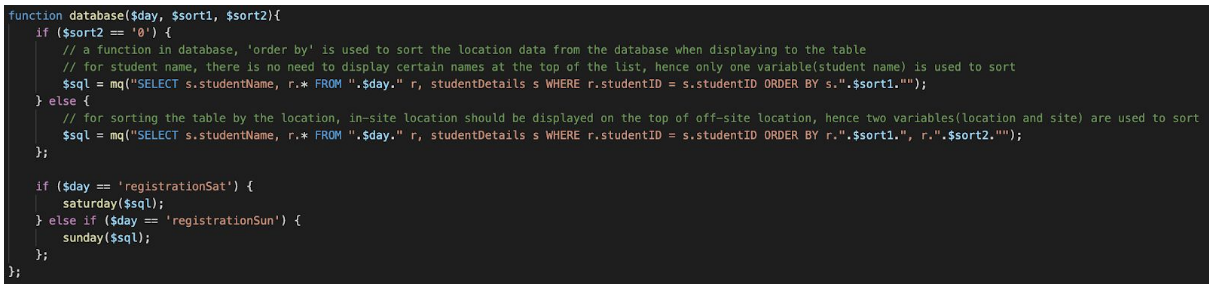
Using while loop, the location data for every student recorded in the database server is printed to the table display. The function ‘saturday’ and ‘sunday’ calls the function ‘display’ according to the day of the week when students entered their location to the system (Figure 8.2).



```php
function saturday($sql) {
        while($row= mysqli_fetch_array($sql)) {
            echo "<tr>";
            echo "<td nowrap=''>" . $row['studentName'] . "</td>";
            display($row['location1'], $row['site1'], $row['time1']);
            display($row['location2'], $row['site2'], $row['time2']);
            display($row['location3'], $row['site3'], $row['time3']);
            echo "</tr>";
        }
        echo "</table>";
}


function sunday($sql) {
    while($row= mysqli_fetch_array($sql)) {
        echo "<tr>";
        echo "<td nowrap=''>" . $row['studentName'] . "</td>";
        display($row['location1'], $row['site1'], $row['time1']);
        display($row['location2'], $row['site2'], $row['time2']);
        echo "</tr>";
    }
    echo "</table>";
}
```

Figure 8.2

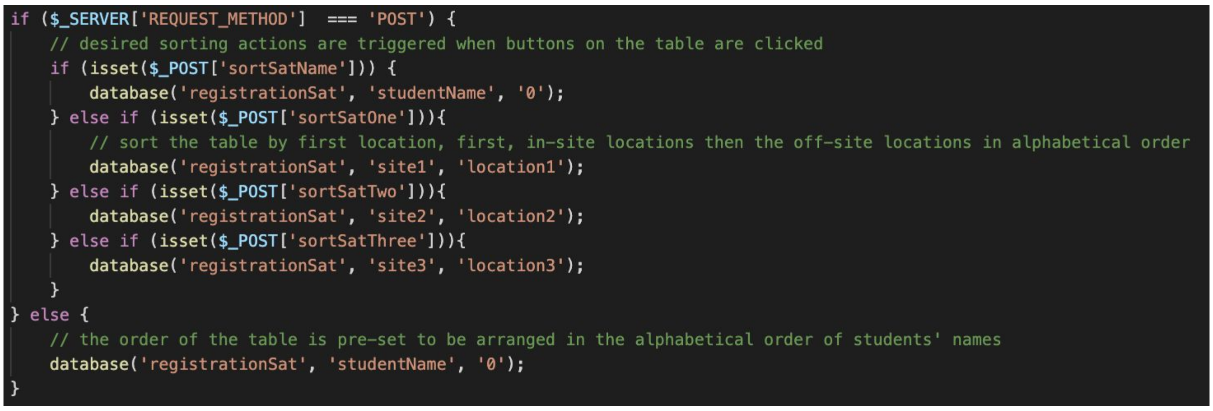
The function ‘saturday’ and ‘sunday’ is called by the function ‘database’, where the query for PHP MySQL for font and background of the location data is decided according to the recorded data from the database (Figure 8.3).



```php
function database($day, $sort1, $sort2){
    if ($sort2 == '0') {
        // a function in database, 'order by' is used to sort the location data from the database when displaying to the table
        // for student name, there is no need to display certain names at the top of the list, hence only one variable(student name) is used to sort
        $sql = mq("SELECT s.studentName, r.* FROM ".$day." r, studentDetails s WHERE r.studentID = s.studentID ORDER BY s.".$sort1."");
    } else {
        // for sorting the table by the location, in-site location should be displayed on the top of off-site location, hence two variables(location and site) are used to sort
        $sql = mq("SELECT s.studentName, r.* FROM ".$day." r, studentDetails s WHERE r.studentID = s.studentID ORDER BY r.".$sort1.", r.".$sort2."");
    };

    if ($day == 'registrationSat') {
        saturday($sql);
    } else if ($day == 'registrationSun') {
        sunday($sql);
    };
};
```

Figure 8.3

Function ‘database’ is called when the user clicks the table headers for sorting the information (Figure 8.4). ‘isset()’ is an embedded function in PHP, which returns boolean value based on the status of the button.

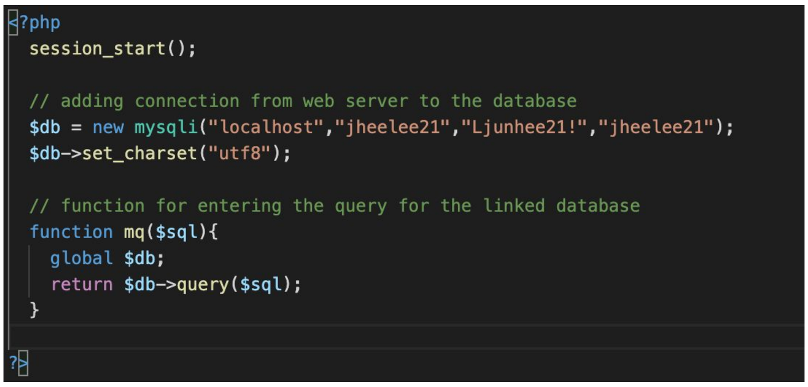


```php
if ($_SERVER['REQUEST_METHOD']  === 'POST') {
    // desired sorting actions are triggered when buttons on the table are clicked
    if (isset($_POST['sortSatName'])) {
        database('registrationSat', 'studentName', '0');
    } else if (isset($_POST['sortSatOne'])){
        // sort the table by first location, first, in-site locations then the off-site locations in alphabetical order
        database('registrationSat', 'site1', 'location1');
    } else if (isset($_POST['sortSatTwo'])){
        database('registrationSat', 'site2', 'location2');
    } else if (isset($_POST['sortSatThree'])){
        database('registrationSat', 'site3', 'location3');
    }
} else {
    // the order of the table is pre-set to be arranged in the alphabetical order of students' names
    database('registrationSat', 'studentName', '0');
}
```

Figure 8.4

1220 words

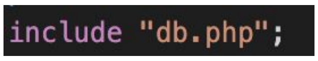
Moreover, ‘db.php’ helps the code for the whole program to be abstract and simple. It maintains the session using ‘session_start()’ which is a pre-written function in php. Hence, there is no need for connecting sessions to the database server every single time when the information from the database is required.



```php
<?php
  session_start();

  // adding connection from web server to the database
  $db = new mysqli("localhost","jheelee21","Ljunhee21!","jheelee21");
  $db->set_charset("utf8");

  // function for entering the query for the linked database
  function mq($sql){
    global $db;
    return $db->query($sql);
  }

?>
```

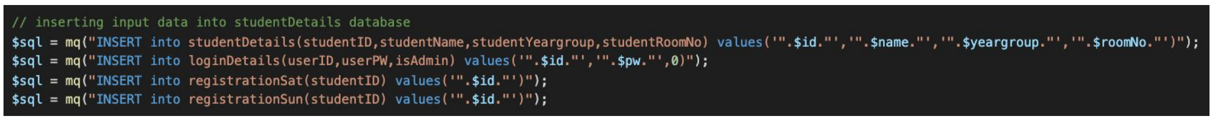
Figure 8.5

The function ‘mq()’ can be used once it is called (Figure 8.6) and it is frequently used in the code, for example in figure 8.7, ‘mq()’ is used to add new student information to the database server.



```php
include "db.php";
```

Figure 8.6



```php
// inserting input data into studentDetails database
$sql = mq("INSERT into studentDetails(studentID,studentName,studentYeargroup,studentRoomNo) values('".$id."','".$name."','".$yeargroup."','".$roomNo."')");
$sql = mq("INSERT into loginDetails(userID,userPW,isAdmin) values('".$id."','".$pw."',0)");
$sql = mq("INSERT into registrationSat(studentID) values('".$id."')");
$sql = mq("INSERT into registrationSun(studentID) values('".$id."')");
```

Figure 8.7