**CS6301.0U1- Web Programming Languages**

**(By Amol Vaze – Net-id: - asv130130@utdallas.edu)**

**Assignment- No-4 Readme Documentation**

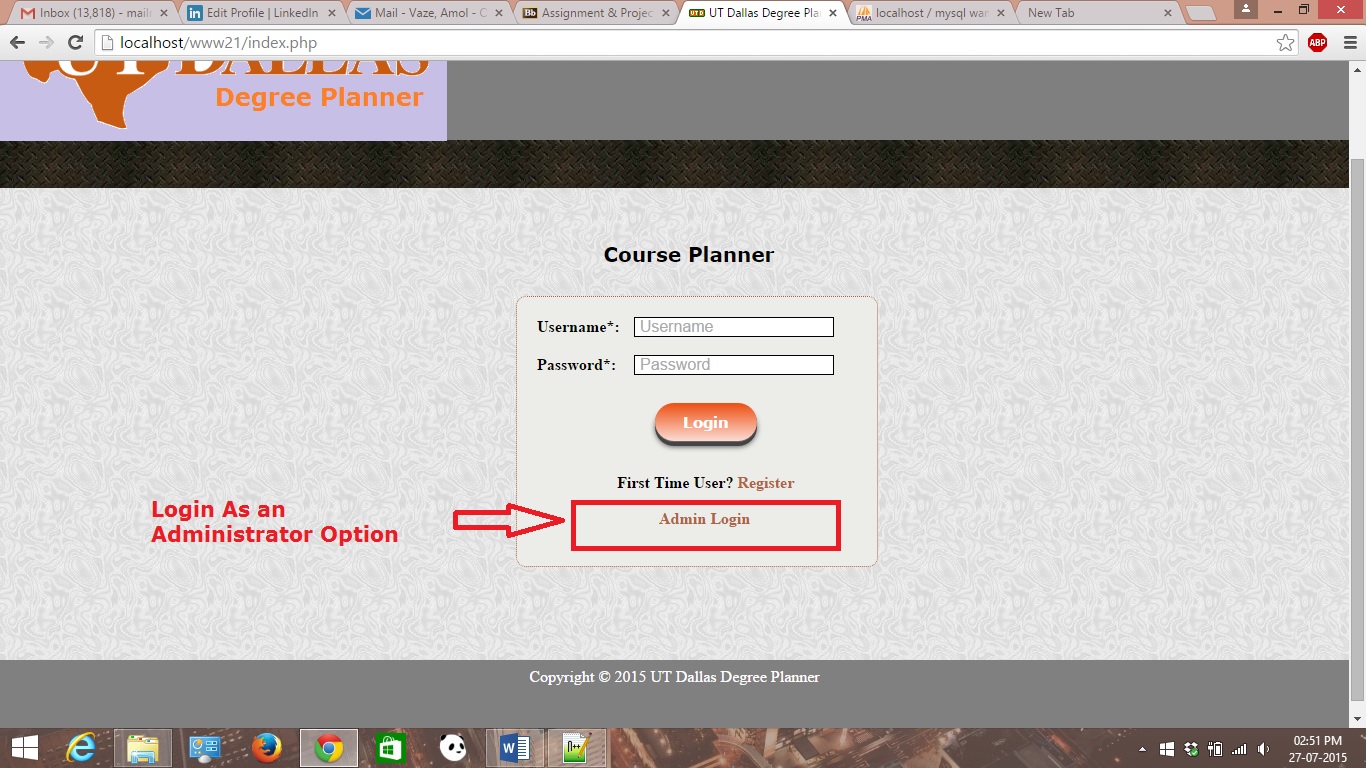
* **Assignment No-4** focuses on web server administration of fully functional UTD degree course planner system for all 6 tracks given.
* My degree planner GUI web page design consists of following files which I have created under a new directory called **www21** under WAMP server (localhost).
* Link to access my local system:- **http://localhost/www21/index.php**
* User should be able to select multiple tracks in this system.
* As a part of third homework extension of homework-1,2 &3, I have created following PHP files which have **HTML** code:-

1. Index.php –[Gives Admin Login option to Login as Admin]
2. Admin.php
3. Admin-profile.php
4. Logout.php [Admin can logout of the system with this logout function.]
5. Usermgmt.php
6. Forgotpwd.php
7. Manageusers.php
8. Update\_user.php
9. Delete-user.php
10. UsabilityTestmgmt.php
11. Userlist.php
12. USE.php

Detailed working & functionalities of this homework can be explained in step by step manner as follows:-

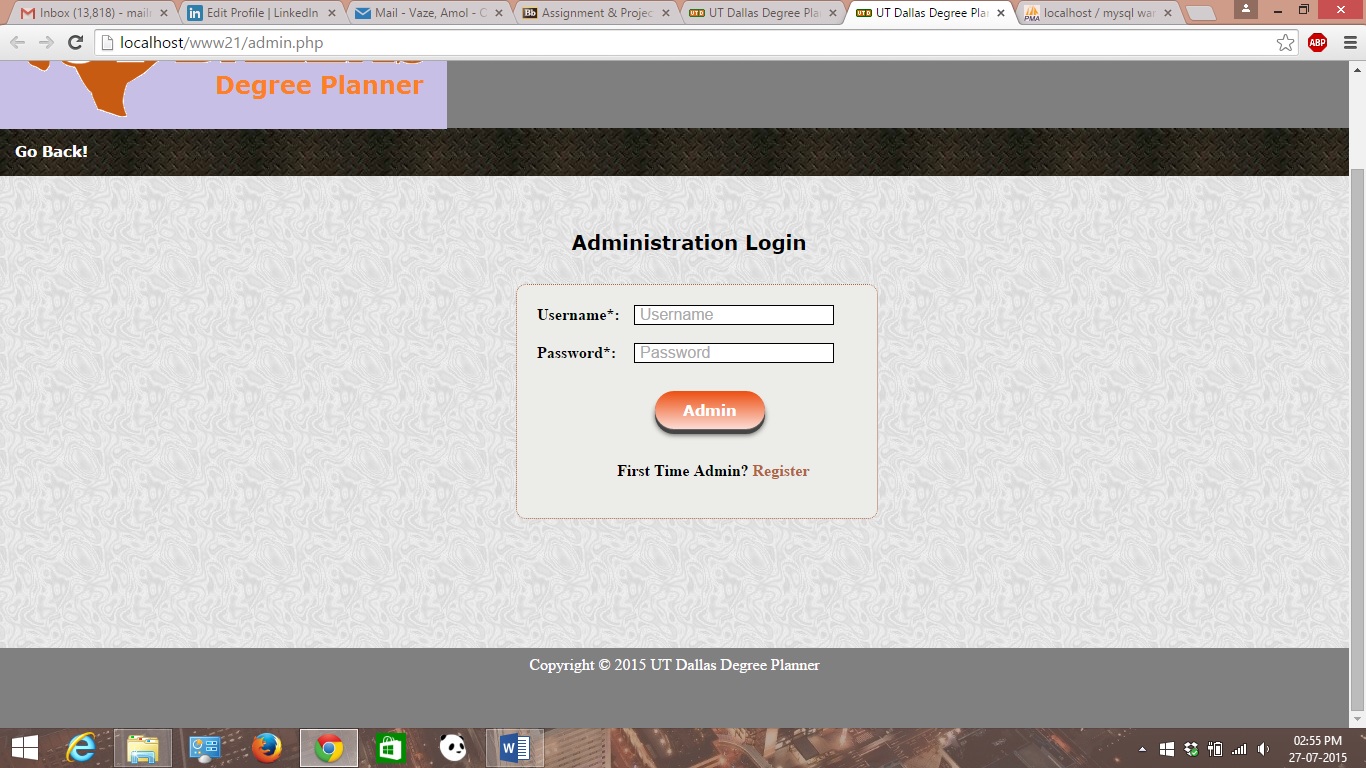
* Home Page or Index.php file:- it is a home page of the system. In case for admin to login as administrator, this login page form has a link at the bottom which facilitates admin login unlike normal user’s login into the system.

A screenshot for admin Login link is attached below:-

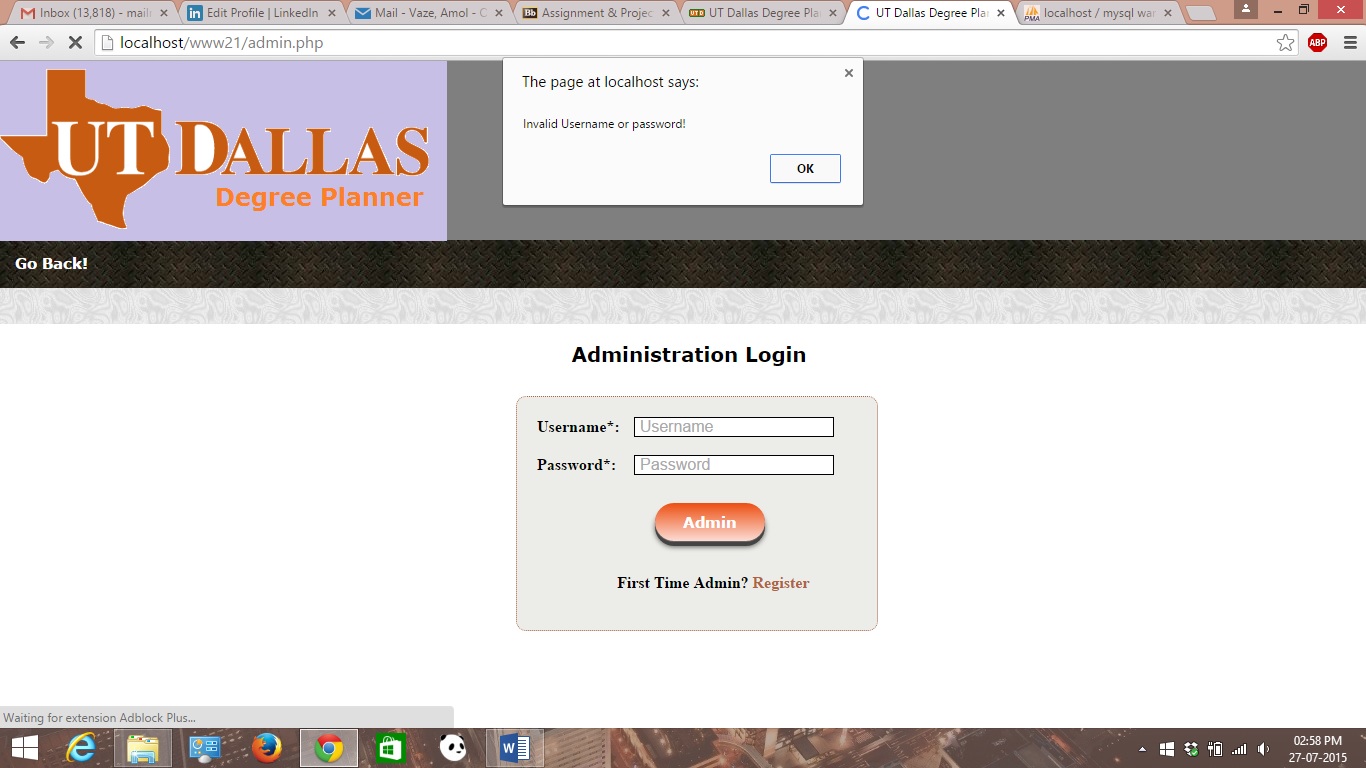


* Admin.php:- It is a separate web page after clicking on admin link in the previous step. This gives the admin user to login into the system to perform the administration functions.
* This page has a register link at the bottom which is taking care of the first time admin user’s into the system like normal login users.

A screenshot for admin-profile page is shown.

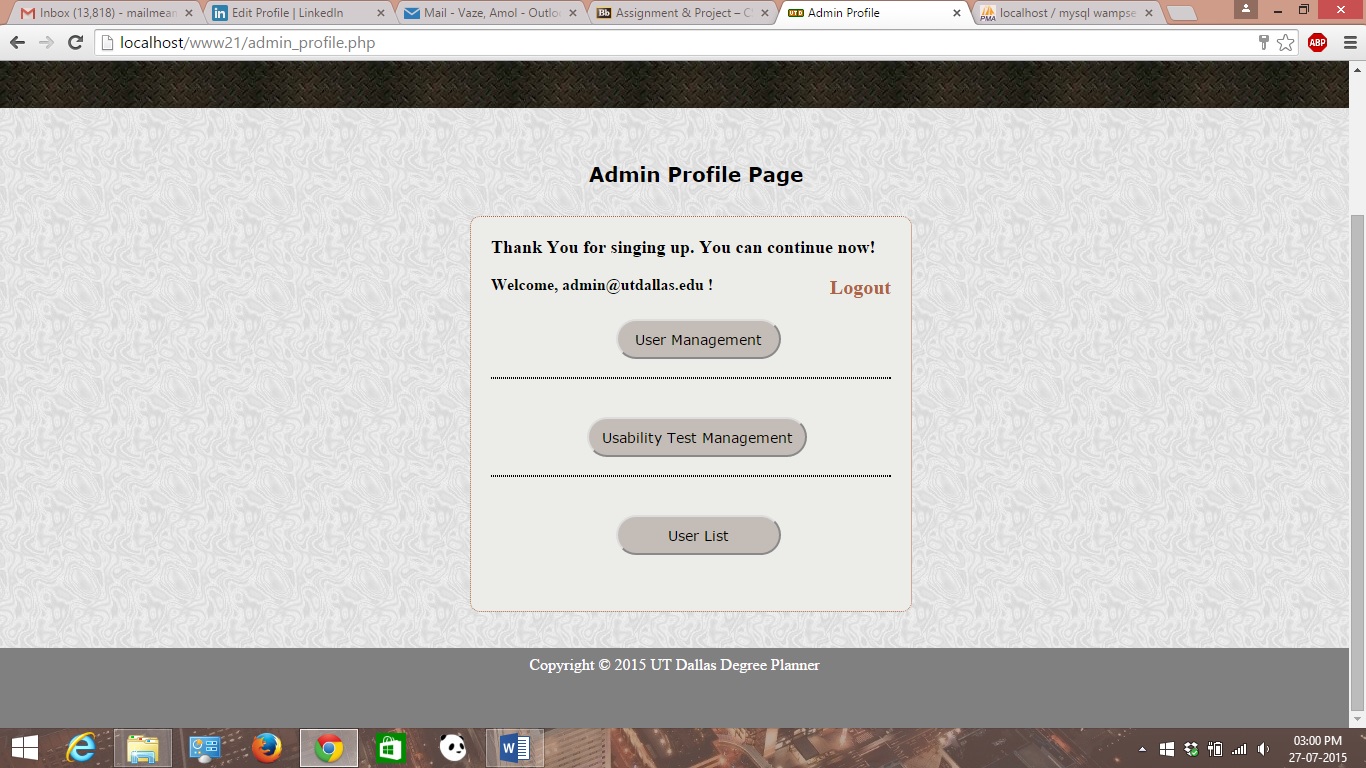


Admin Validation has been achieved using client side JavaScript as shown.



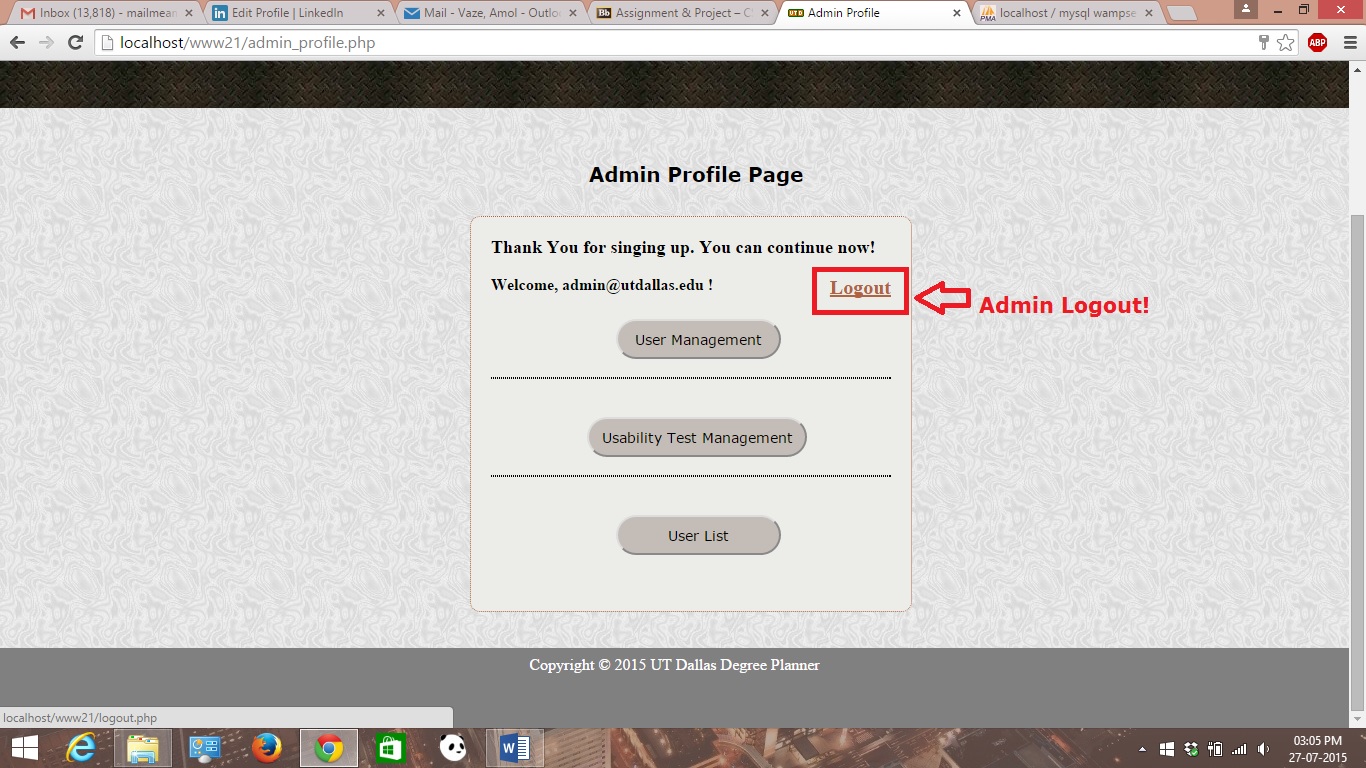
* Admin-profile page: - It is a web page which shows the admin profile once admin has logged in into the system. It then shows various set of tasks that admin could do as an admin user like user mgmt., user test mgmt. and user lists respectively.

A screenshot is attached below:-



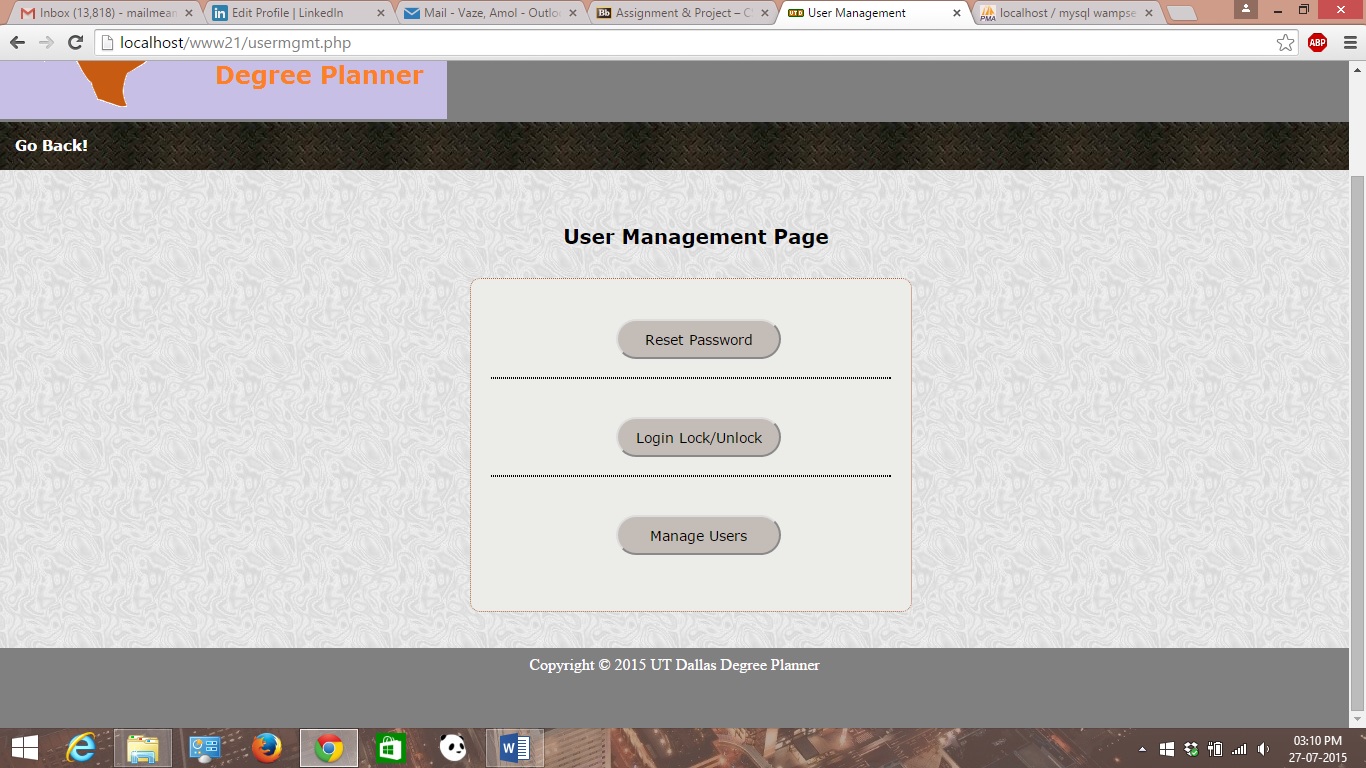
* Admin-Logout.php:- It is a file which makes admin logout of the system. Once admin has clicked this button, he should be able to log out of the system & session for that particular admin user will get destroyed.

A screenshot for the logout is attached below:-



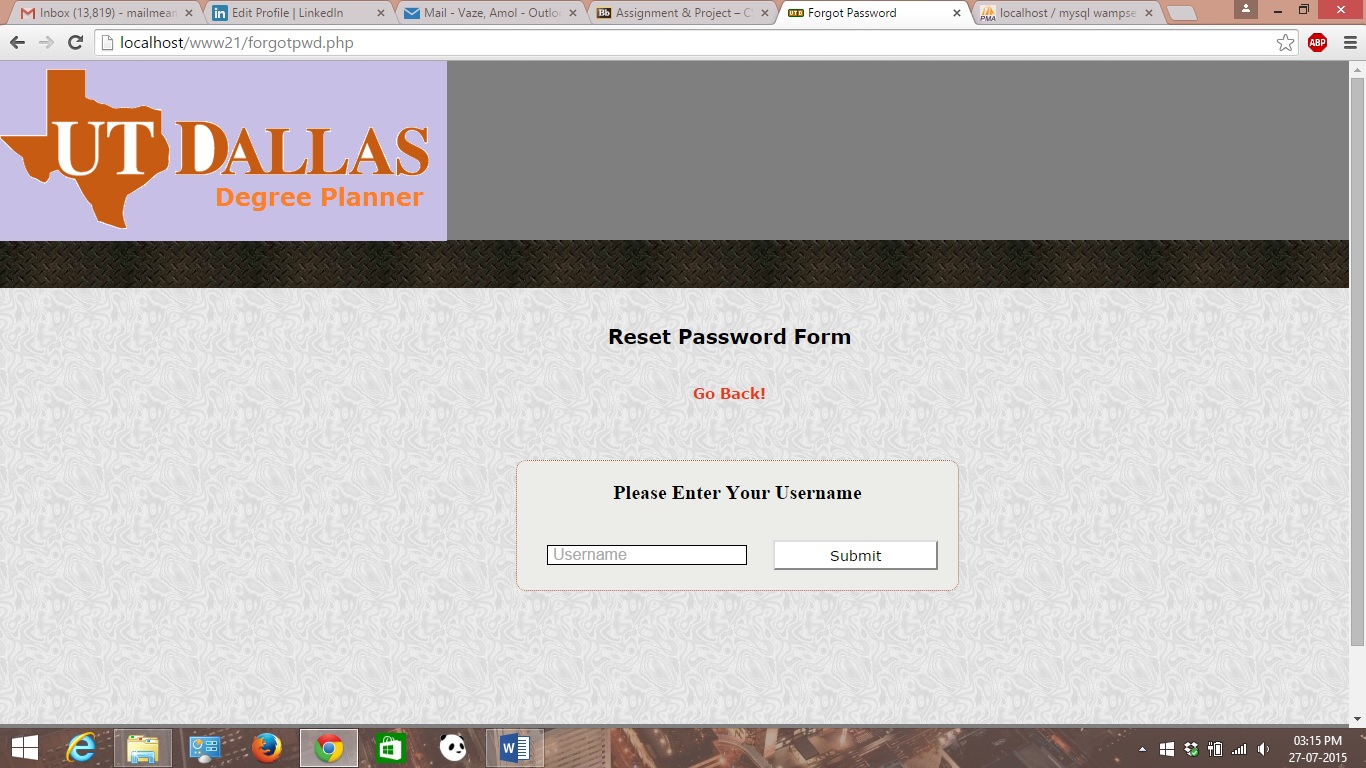
* Usermgmt.php:- It is a separate web page showing user management functions that admin could perform on. It basically involves functions like reset password, login lock/unlock and manage users respectively.

A screenshot is attached below.

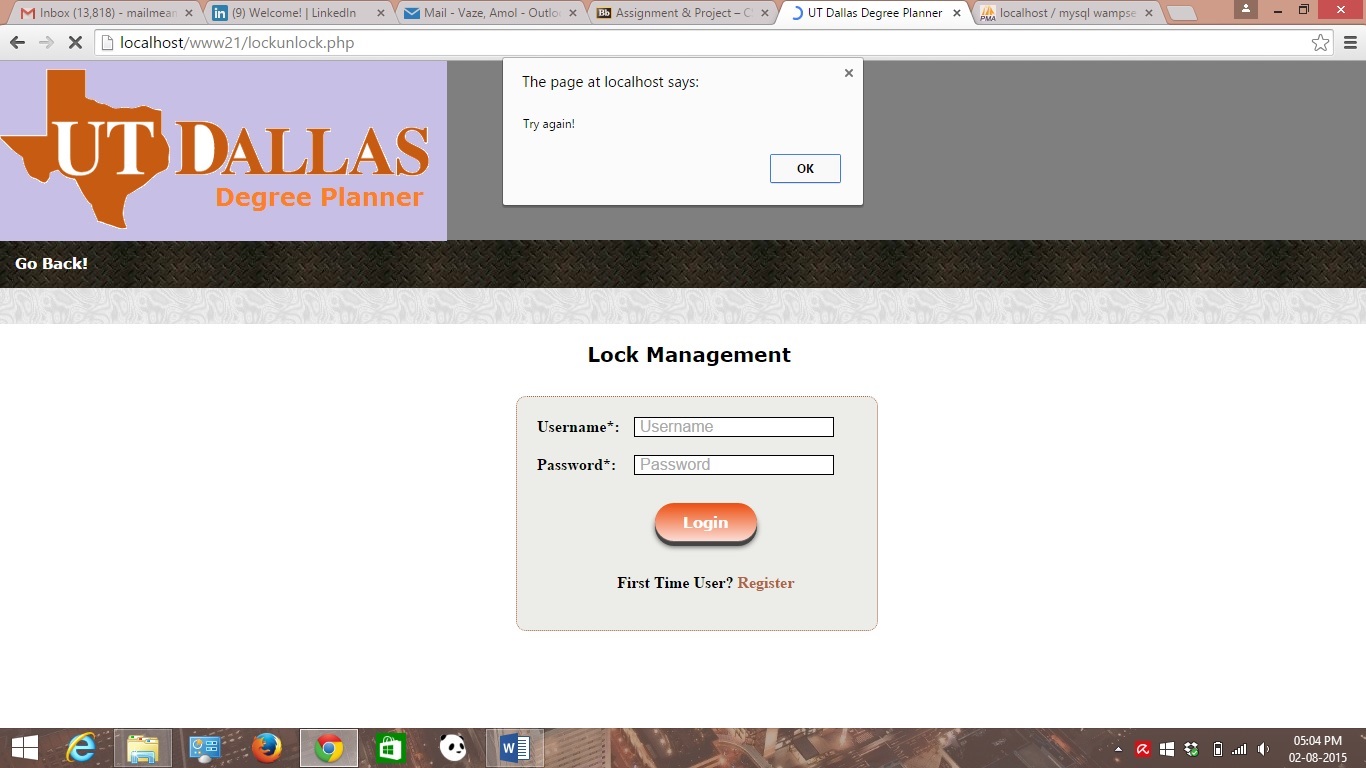


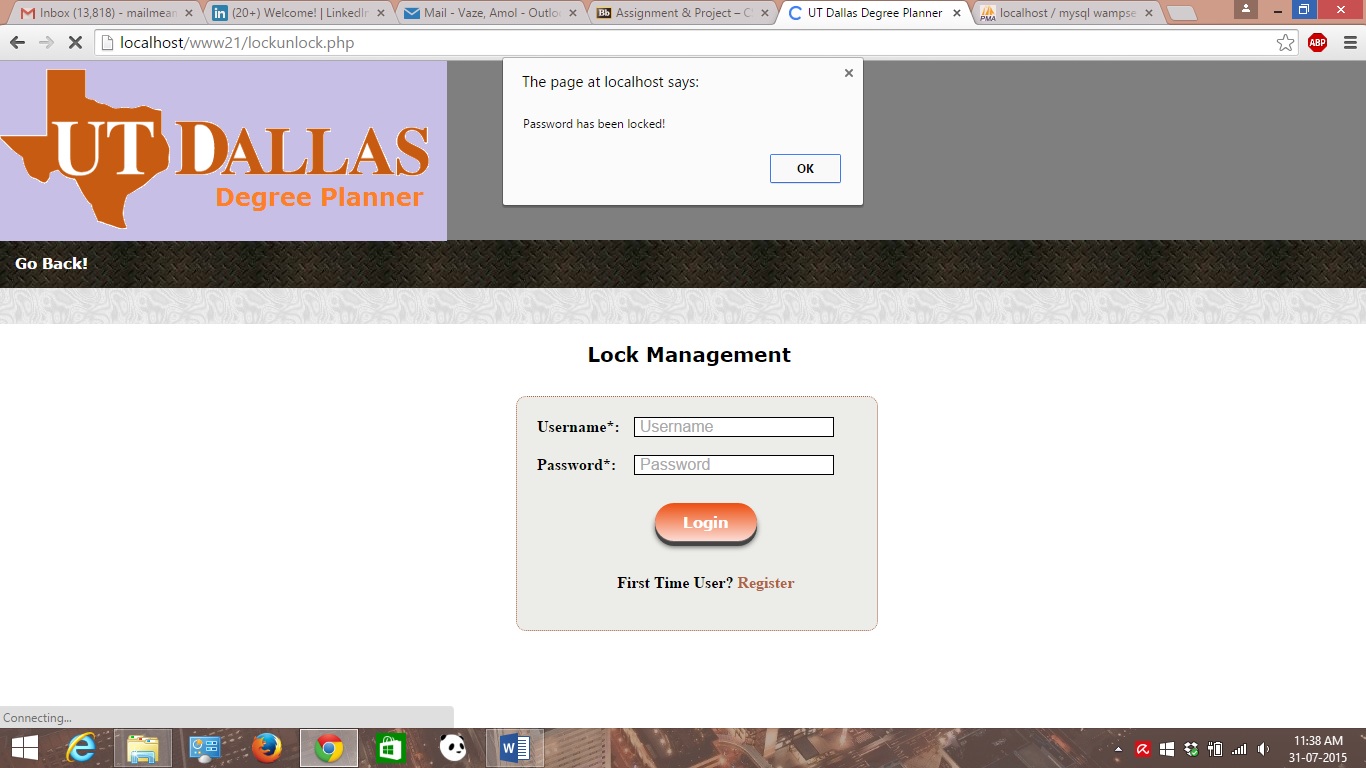
* Reset password includes forgot password page which includes sending user resettled password by taking user’s id as an input.

A Screenshot is attached here.



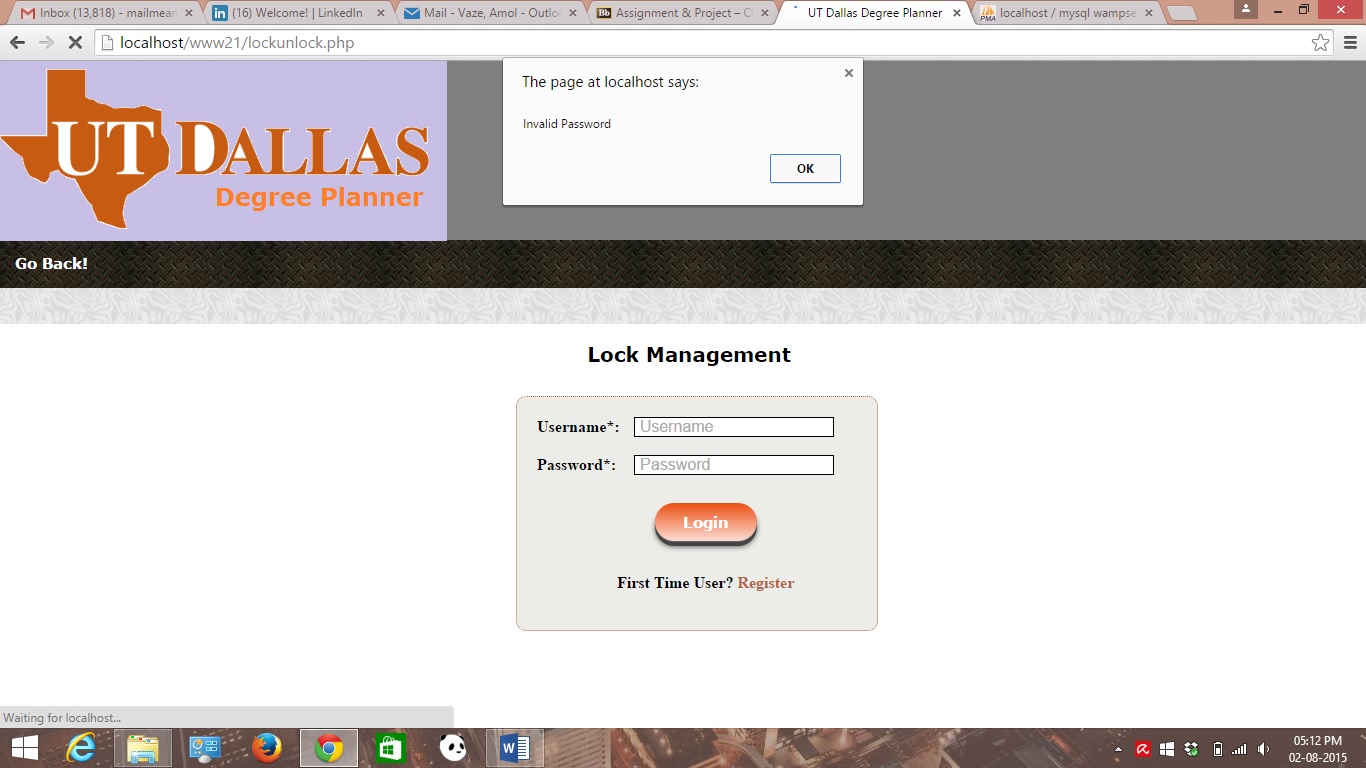
* Login lock/unlock: It is a web page on clicking of it gets redirected to the home lock/unlock management page of the system & it checks whether the user is entering correct credentials or not. If not it won’t let the users proceed & system first displays the message “Try Again” which in turn locks the user account. A account locked message will be displayed followed by Try again message. Following screenshots represent locking management of user’s account is shown below.



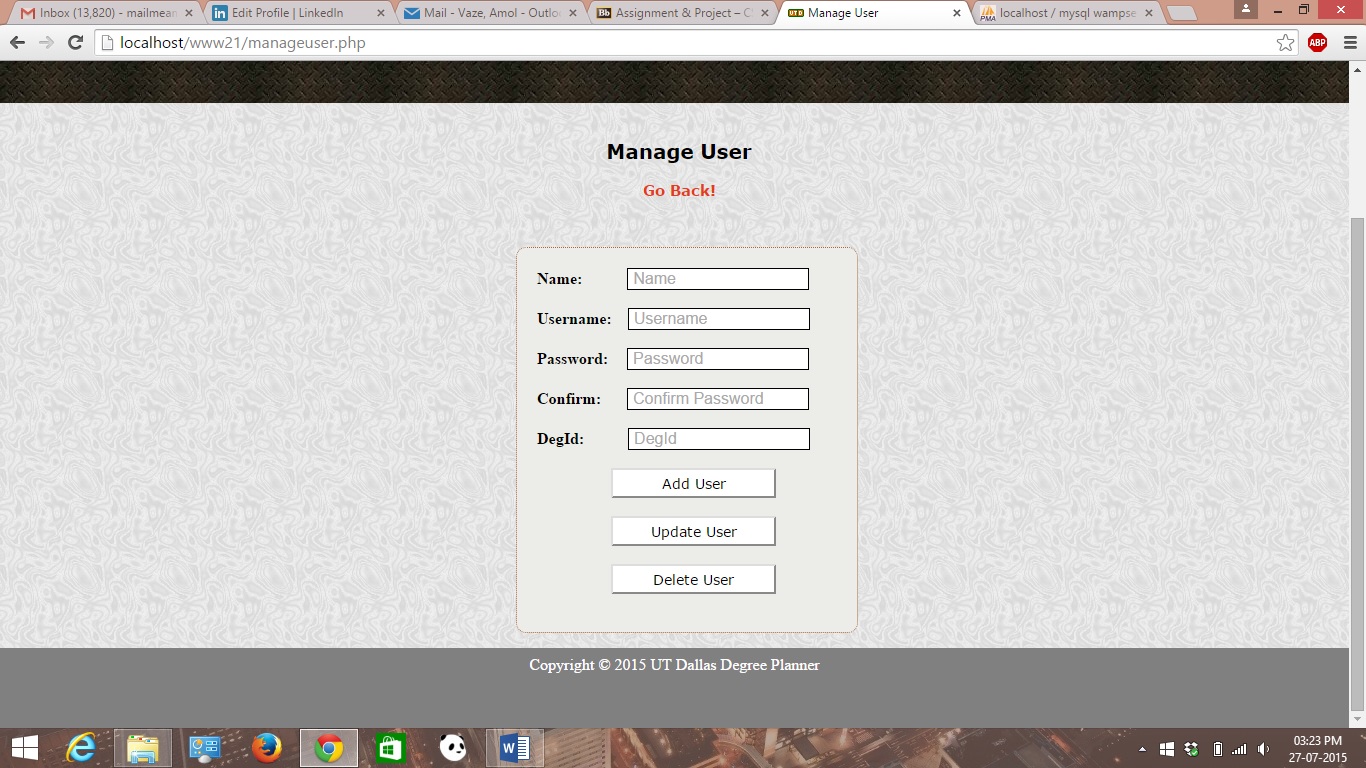


Finally one more case has been handled in case of locking mechanism where when the user is not present in the database table & suppose that user is outside user. In that if outside user tries to access the system, system will restrict the access by displaying the message prompt saying that invalid password.

A screenshot for this case is shown below.

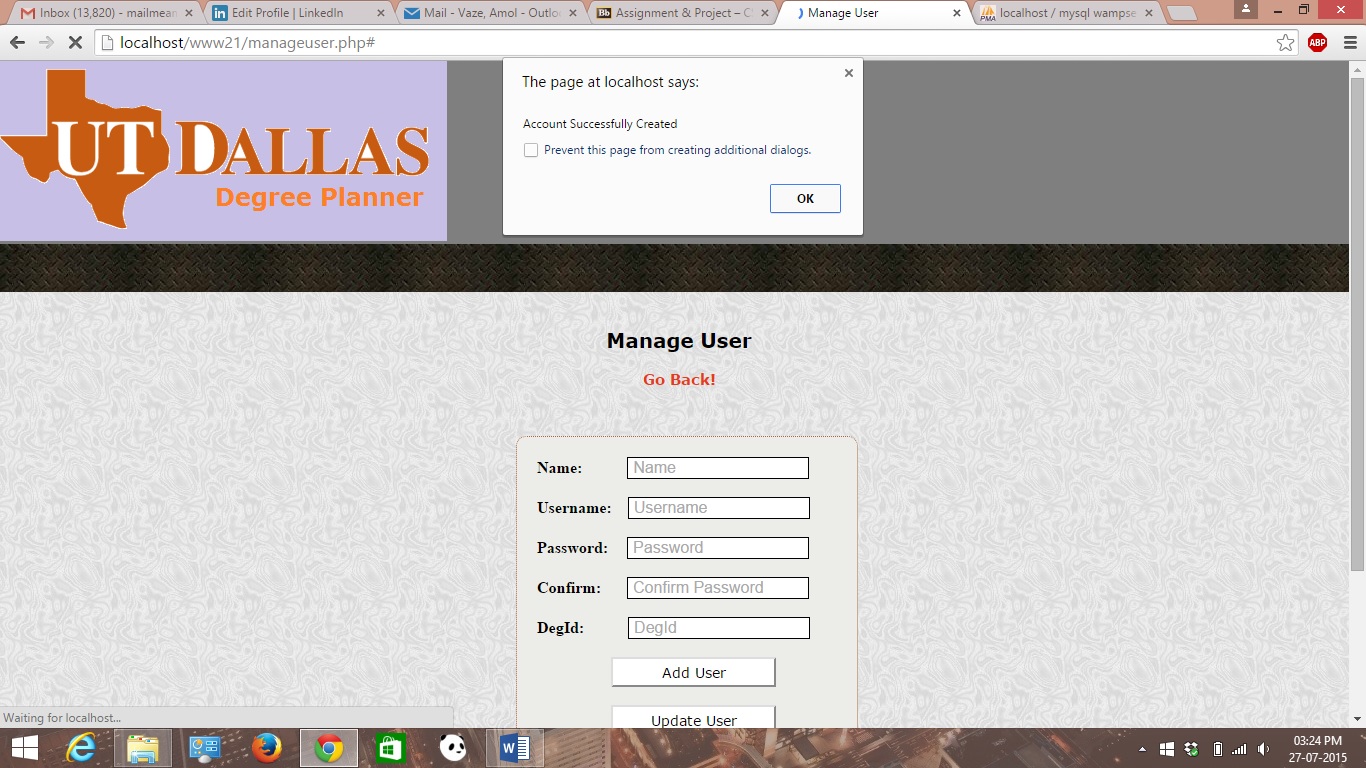


* Manage User’s- It is a web page which helps admin user to do user management. Admin should be able to add, update and delete existing user’s info with this form.

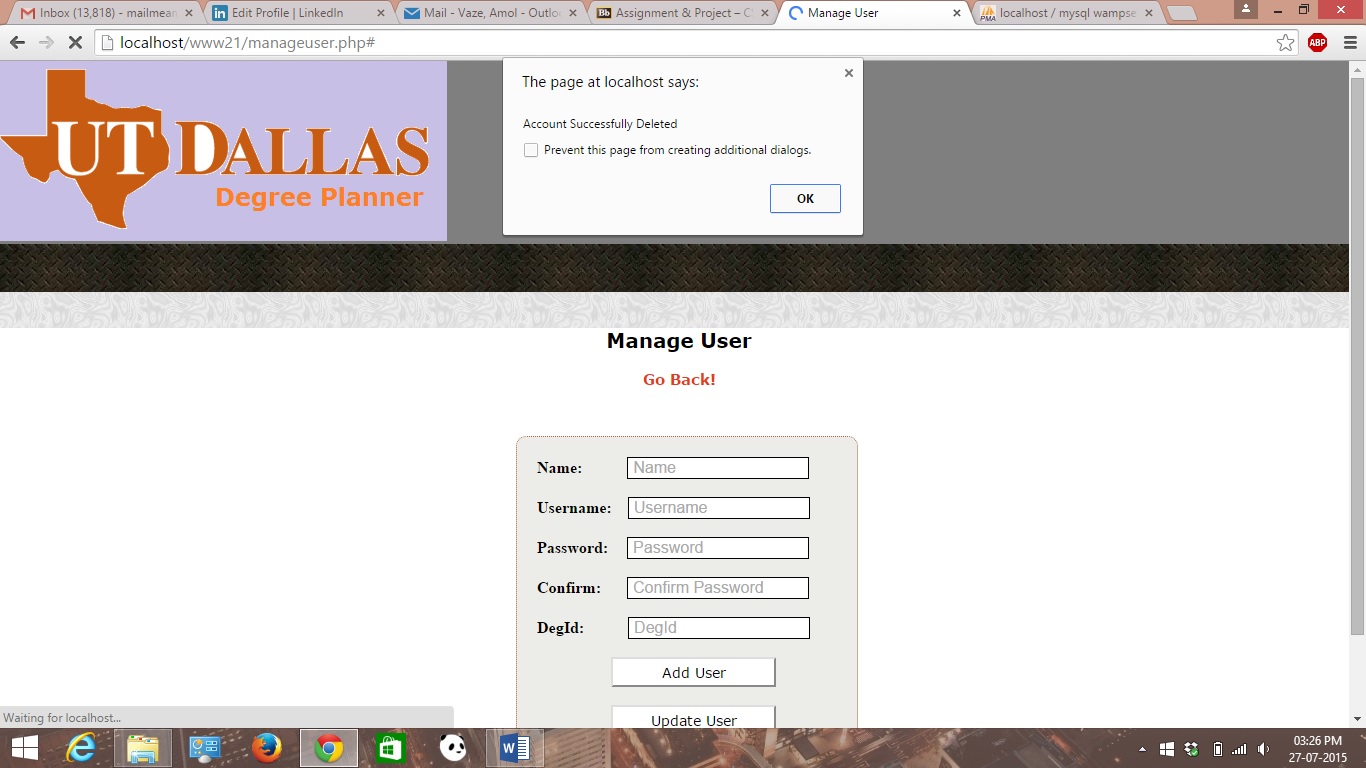


An evidence for the above admin functionality can be shown as below:

1. When admin adds new user into the system, new user account will get created into the user’s table. A screenshot for the same is shown. This can be achieved by the manageusers.php

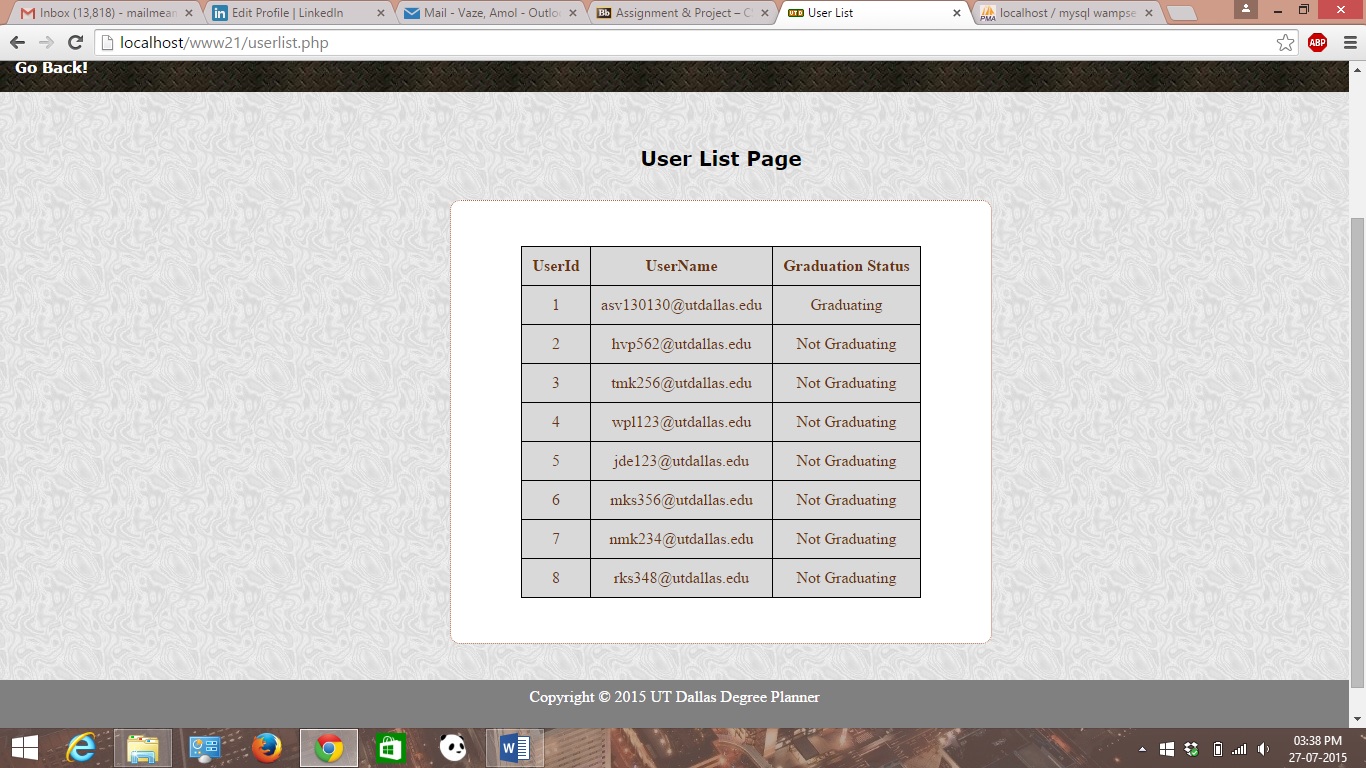


1. When admin deletes existing user from the system, then account has been deleted for that particular user. This can be achieved by the code in delete\_user.php



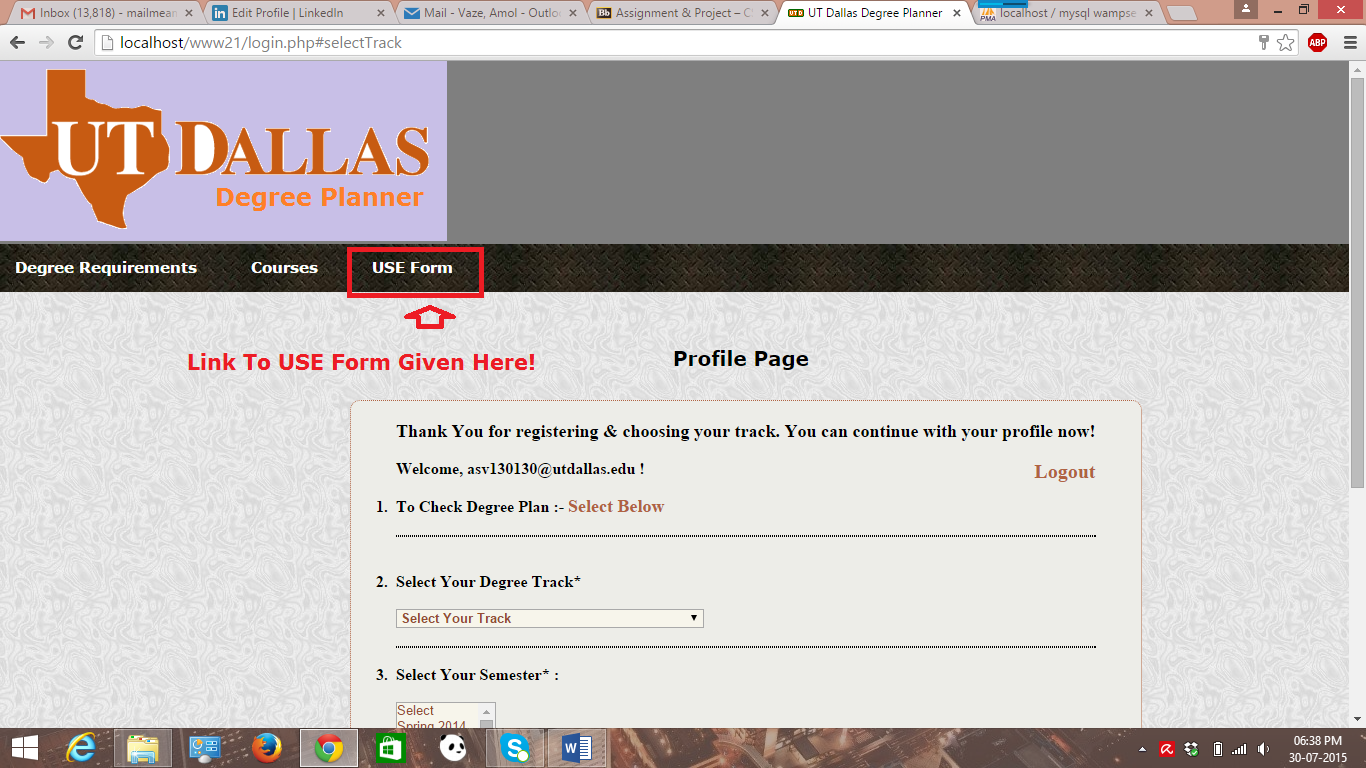
* Userlist.php is a web page which displays the current student’s status from the database table for regarding the graduation based on the degree plan taken by that student user.
* For example, in the screenshot attached, it shows first user is graduating while rest are not graduating this semester.
* The user list displayed contains the information of user id, username and graduation status of the students respectively.

A screenshot is attached below:-



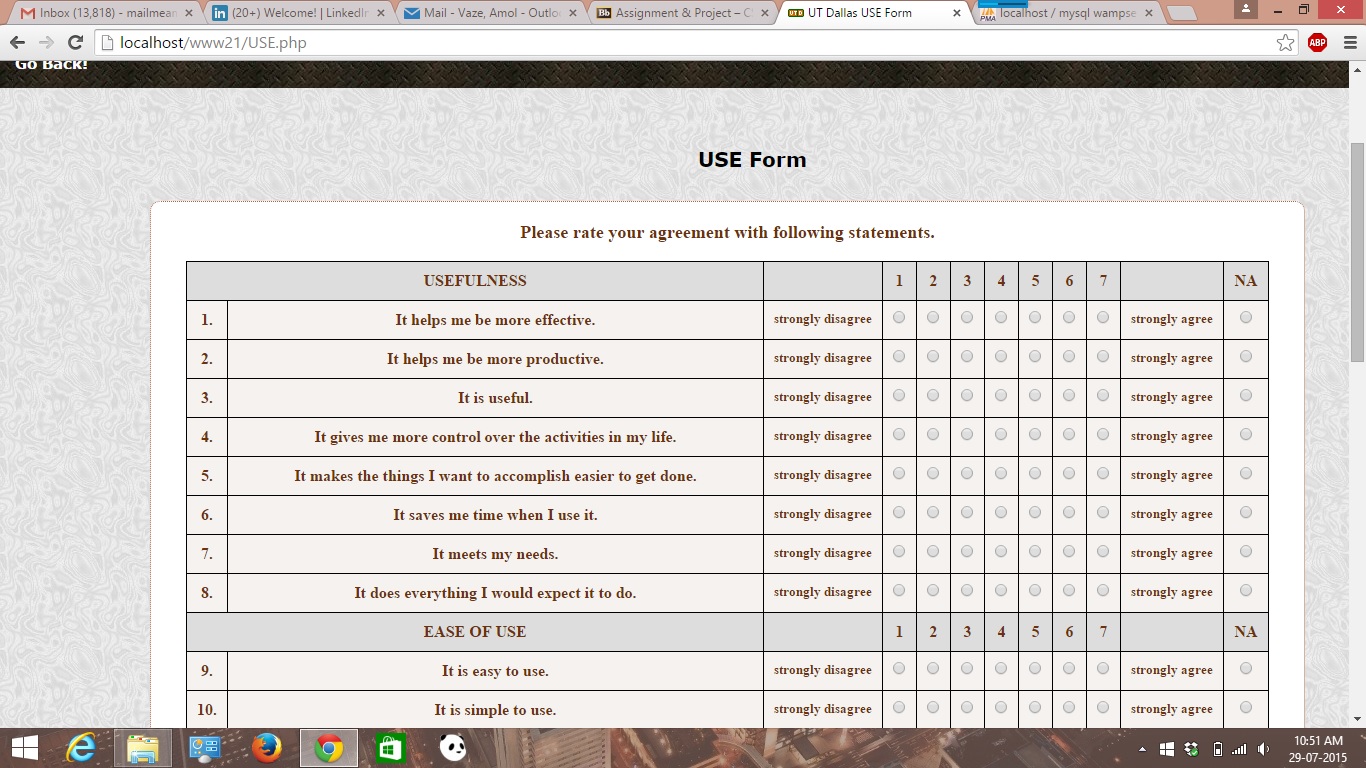
* USE.php :- From Login page of the website, user should be able to navigate to USE form in the system. In this way a user should be able to access USE survey form inside the system.

A screenshot showing USE Form link can be shown as follows:-

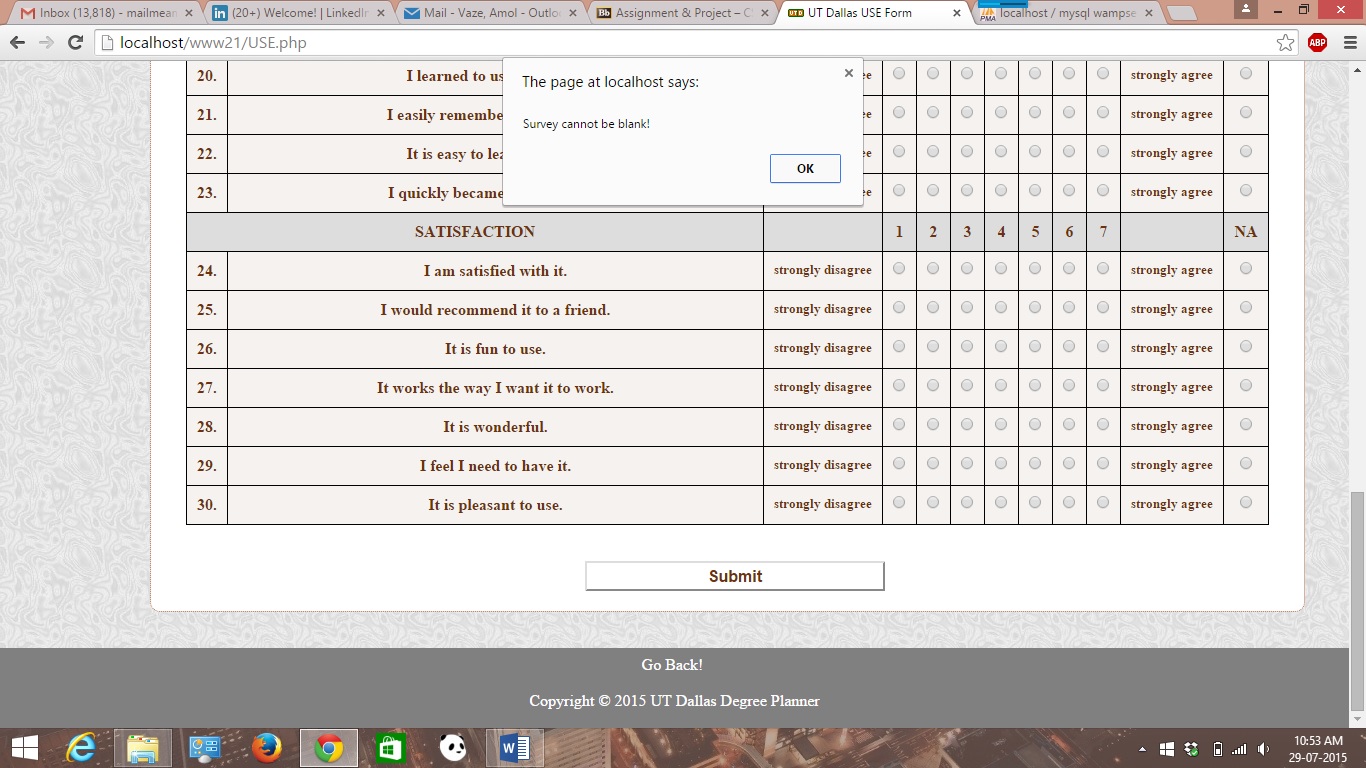


* It is a web page which has survey form and submit button to save the details of the survey. This form has a set of radio buttons for the different questions that need to be answered while filling out the survey form.

A screenshot for USE form has been attached below.

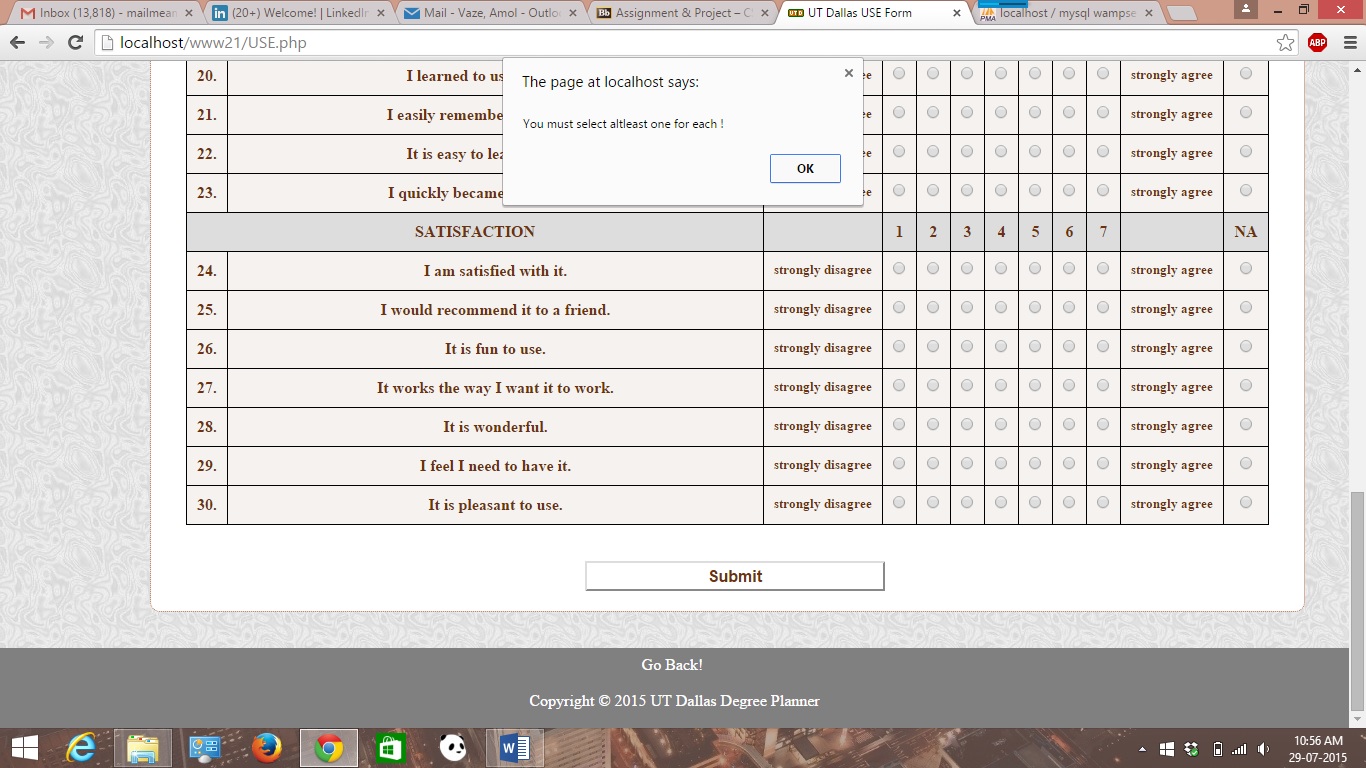


USE Form has validation controls done using JavaScript. Whenever user tries to submit blank form, it shows corresponding popup. A screenshot of the same is attached below.

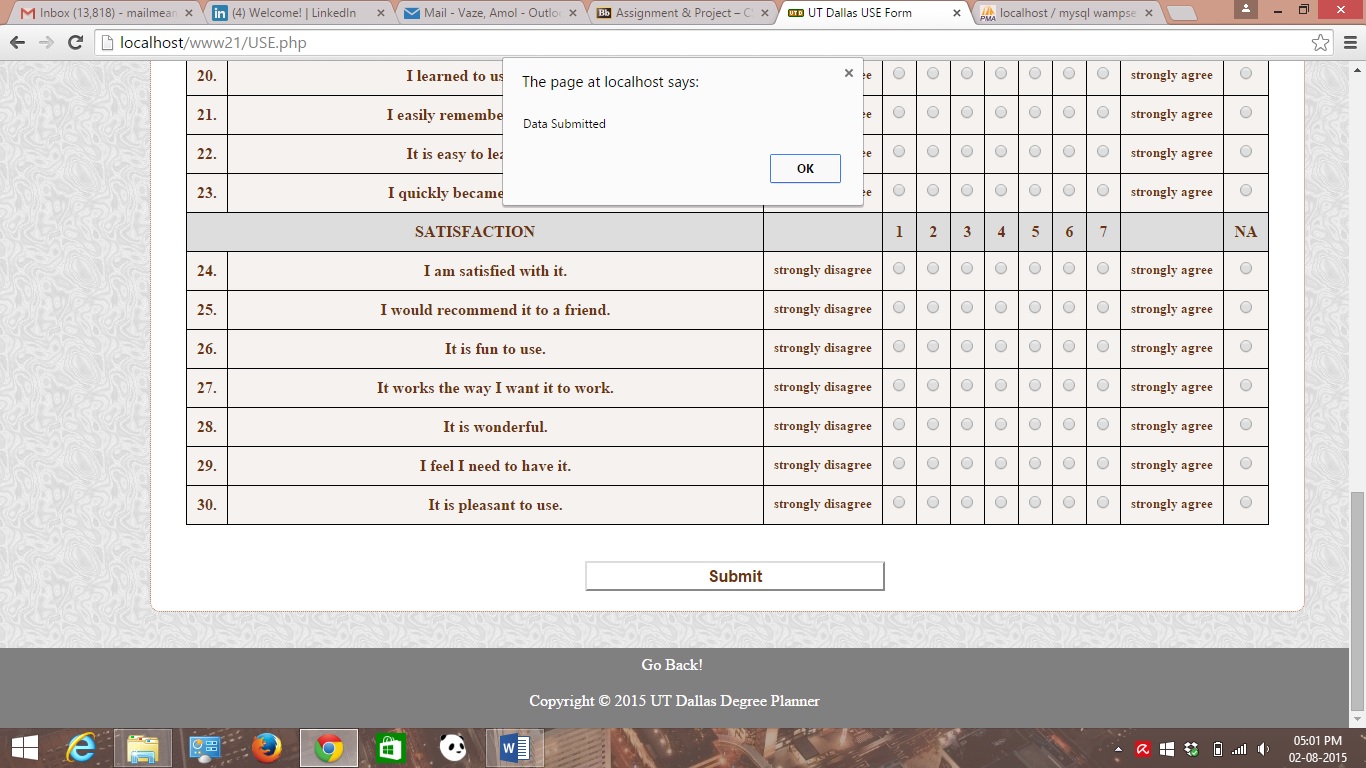


In addition to blank form validation, USE from does the validation like it won’t allow user to leave any individual radio button field blank.

A screenshot of this validation is shown below:-



In case of USE form when user submits the form it will show that data submitted prompt to the user. A screenshot for the same is attached below.



* Regarding cookie and session for each user logged into system a piece of code has been used where PHP session and cookie variables are used inside PHP script in order to satisfy the requirements for both.
* A piece of code for maintaining **user session** inside Login PHP script can be shown as follows:-

// Few pieces of codes representing the concept of session for each user’s session using PHP session variables.

1. **session\_start**();

if(!isset**($\_SESSION**["sess\_user"])){

header("location:login.php");

}

1. Welcome, <?=**$\_SESSION**['sess\_user'];?> ! <a href="logout.php" id = "logout">Logout</a>

* Regarding **cookies for the system in PHP**, following piece of code has been put into Login PHP script in order to maintain cookies for each user.

// Login script contains a piece of code representing cookie for the PHP system.

$**cookie\_name** = "user";

$cookie\_value = $\_SESSION ['sess\_user'];

// setting cookie for each user

setcookie($cookie\_name, $cookie\_value, strtotime( '+30 days' ), "/", "", "", TRUE);

//echo "Cookie '" . $cookie\_name . "' is set!<br>";

//echo "Cookie Value is: " . **$\_COOKIE** [$cookie\_name];

* Thus, above code pretty much explains the use of PHP cookies where a cookie name is mandatory. Also, this helps maintain cookie for every user that has been logged in into the system.
* Like we have to set both cookie and session variables, once the user leaves the system, we have to unset and or destroy the same. Following piece of code has been put into logout.php for un setting and or destroying session and cookie.

**Unset** ($\_SESSION['sess\_user']);

// destroying cookie

**setcookie**("user", $\_SESSION['sess\_user'], strtotime( '**-5 days'** ), '/');

// destroying session

**session\_destroy**();

* For un-setting cookie **-5 days** has been used instead of positive value.
* **P.S:-** [In case of cookies, few lines of code has been commented out. That is because for our understanding purpose like echoing out the results to see cookie values and cookie name are properly set or not.]
* All database tables used for the system have been attached in separate SQL file which has a set of all the tables with their schema. It also includes all the necessary insert table statements for populating database tables with the data needed.
* Also, during the submission, separate header.php and footer.php files have been used as per the question asked in homework.

**Database Tables**

**Admin Table**

CREATE TABLE IF NOT EXISTS `admin` (

`Id` smallint(6) NOT NULL AUTO\_INCREMENT,

`Name` varchar(50) NOT NULL,

`Username` varchar(32) NOT NULL,

`Password` varchar(32) NOT NULL,

`DegId` smallint(6) DEFAULT NULL,

PRIMARY KEY (`Id`),

KEY `DegId` (`DegId`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=3 ;

**Class Table**

CREATE TABLE IF NOT EXISTS `class` (

`ClassId` int(11) NOT NULL,

`ClassName` varchar(100) DEFAULT NULL,

`ClassPrfx` char(4) DEFAULT NULL,

`ClassHours` int(11) DEFAULT NULL,

`DegId` smallint(6) DEFAULT NULL,

`Grade` smallint(6) DEFAULT NULL,

PRIMARY KEY (`ClassId`),

KEY `DegId` (`DegId`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

**Courses Table**

CREATE TABLE IF NOT EXISTS `courses` (

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

`number` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL,

`prefix` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL,

`name` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL,

`description` text COLLATE utf8\_unicode\_ci,

`created\_at` datetime NOT NULL,

`updated\_at` datetime NOT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci AUTO\_INCREMENT=73 ;

**DegreePlan Table**

CREATE TABLE IF NOT EXISTS `degreeplan` (

`DegId` smallint(6) NOT NULL AUTO\_INCREMENT,

`DegName` varchar(50) DEFAULT NULL,

`DeptId` smallint(6) DEFAULT NULL,

PRIMARY KEY (`DegId`),

KEY `DeptId` (`DeptId`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=7 ;

**Department Table**

CREATE TABLE IF NOT EXISTS `department` (

`DeptId` smallint(6) NOT NULL AUTO\_INCREMENT,

`DeptName` varchar(50) DEFAULT NULL,

`DeptInit` char(4) DEFAULT NULL,

PRIMARY KEY (`DeptId`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=5 ;

**Grade Table**

CREATE TABLE IF NOT EXISTS `grade` (

`GradeId` smallint(6) NOT NULL AUTO\_INCREMENT,

`ClassId` int(11) DEFAULT NULL,

`UserId` smallint(6) DEFAULT NULL,

`Passed` bit(1) DEFAULT NULL,

`Grade` decimal(18,2) DEFAULT NULL,

`GpaWeight` decimal(18,2) DEFAULT NULL,

`GradeLetter` varchar(16) NOT NULL,

PRIMARY KEY (`GradeId`),

KEY `ClassId` (`ClassId`),

KEY `UserId` (`UserId`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=18 ;

**Graduation Table**

CREATE TABLE IF NOT EXISTS `graduation` (

`UserId` smallint(6) NOT NULL AUTO\_INCREMENT,

`Name` varchar(50) NOT NULL,

`Username` varchar(32) NOT NULL,

`Grad\_Status` varchar(32) NOT NULL,

`DegId` smallint(6) DEFAULT NULL,

PRIMARY KEY (`UserId`),

KEY `DegId` (`DegId`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=9 ;

**Prereq Table**

CREATE TABLE IF NOT EXISTS `prereq` (

`ClassId` int(11) NOT NULL,

`ClassName` varchar(100) DEFAULT NULL,

`ClassPrfx` char(4) DEFAULT NULL,

`ClassHours` int(11) DEFAULT NULL,

`DegId` smallint(6) DEFAULT NULL,

`UserId` smallint(6) DEFAULT NULL,

PRIMARY KEY (`ClassId`),

KEY `UserId` (`UserId`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

**Survey Table**

CREATE TABLE IF NOT EXISTS `survey` (

`Id` smallint(6) NOT NULL AUTO\_INCREMENT,

`row1` varchar(50) NOT NULL,

PRIMARY KEY (`Id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=1 ;

**User Table**

CREATE TABLE IF NOT EXISTS `user` (

`UserId` smallint(6) NOT NULL AUTO\_INCREMENT,

`Name` varchar(50) NOT NULL,

`Username` varchar(32) NOT NULL,

`Password` varchar(32) NOT NULL,

`DegId` smallint(6) DEFAULT NULL,

PRIMARY KEY (`UserId`),

KEY `DegId` (`DegId`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=9 ;

**Conclusion**

To conclude, this document gives the thorough overview of the system with screenshots attached and it clearly explains the USE form and administration level functional requirements needed by the system. Thus, this concludes the working of fully functional system up & running for degree planner.

[**Please Note**: Database with all tables, JavaScript & Ajax files required by the code are also submitted/attached with the submission through elearning.]