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

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

**Assignment- No-2 Readme Documentation**

* **Assignment No-2** focuses on design & implementation of fully functional UTD degree course planner system.
* 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**
* Being I am into Data Science track, I am implementing degree planner for the same.
* As a part of second homework extension of homework-1 I have created following php files which have **HTML** code:-

1. Register.php
2. Login.php- (Showing [Initial user’s] MyProfile )
3. Logout.php
4. Forgot.php (Showing forgot password page)
5. Degree-plan.php
6. Grade.php
7. Add-classes.php(CURD- ADD)
8. Drop-classes.php(CURD- Drop)
9. Update-classes.php(CURD- Update)

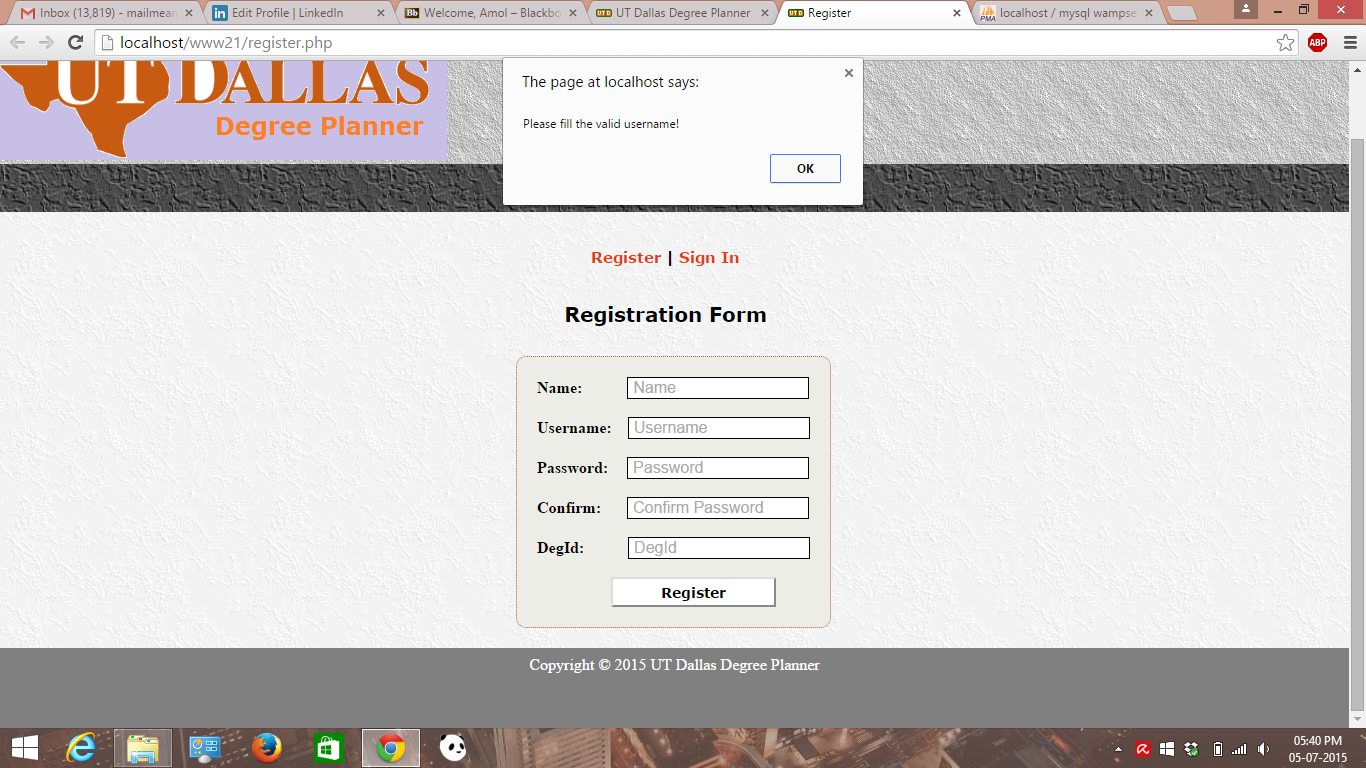
**Detailed Description of each above as follows:-**

**​Screen shots with explanation (what is going on) for each step**

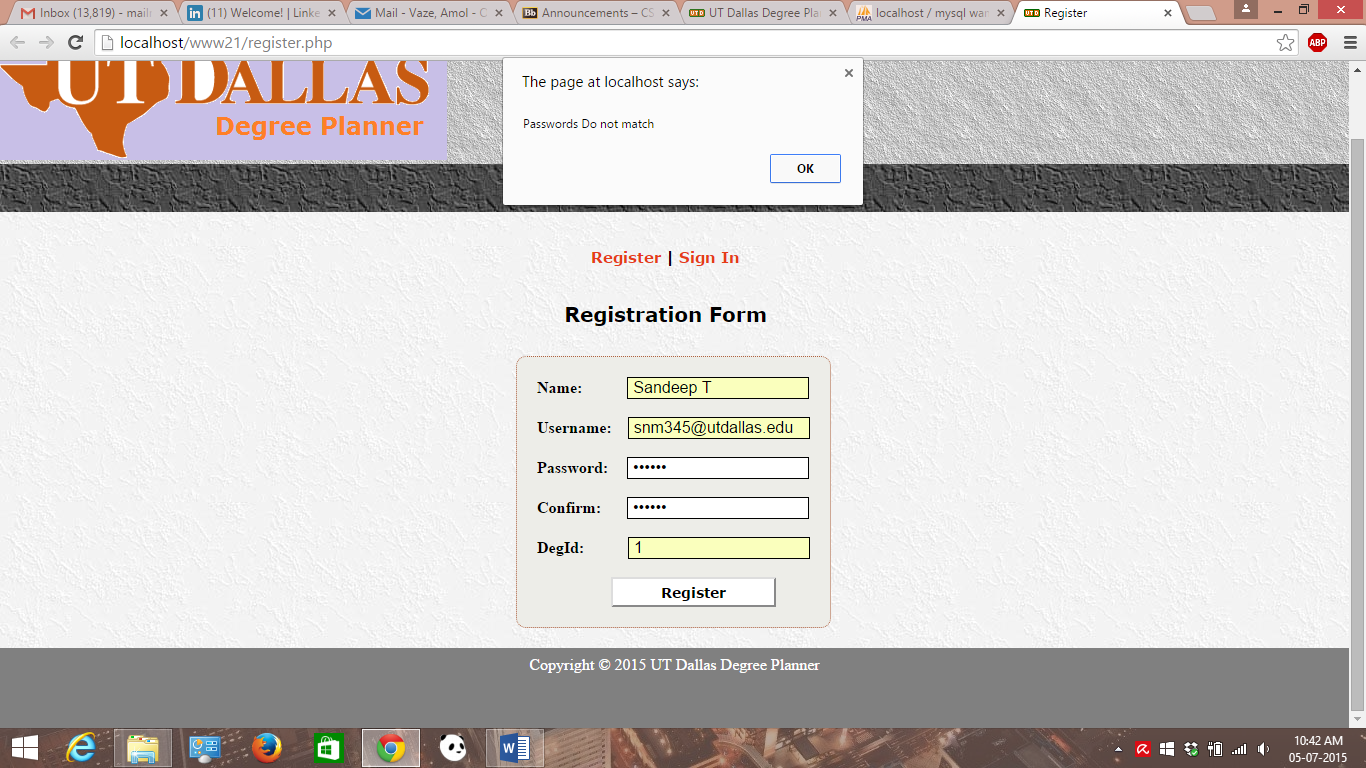
* **Register.php is** a web page which has form for registration & it is basically used when new student user is being added to the system. It will take details from GUI form and enters into the database table.



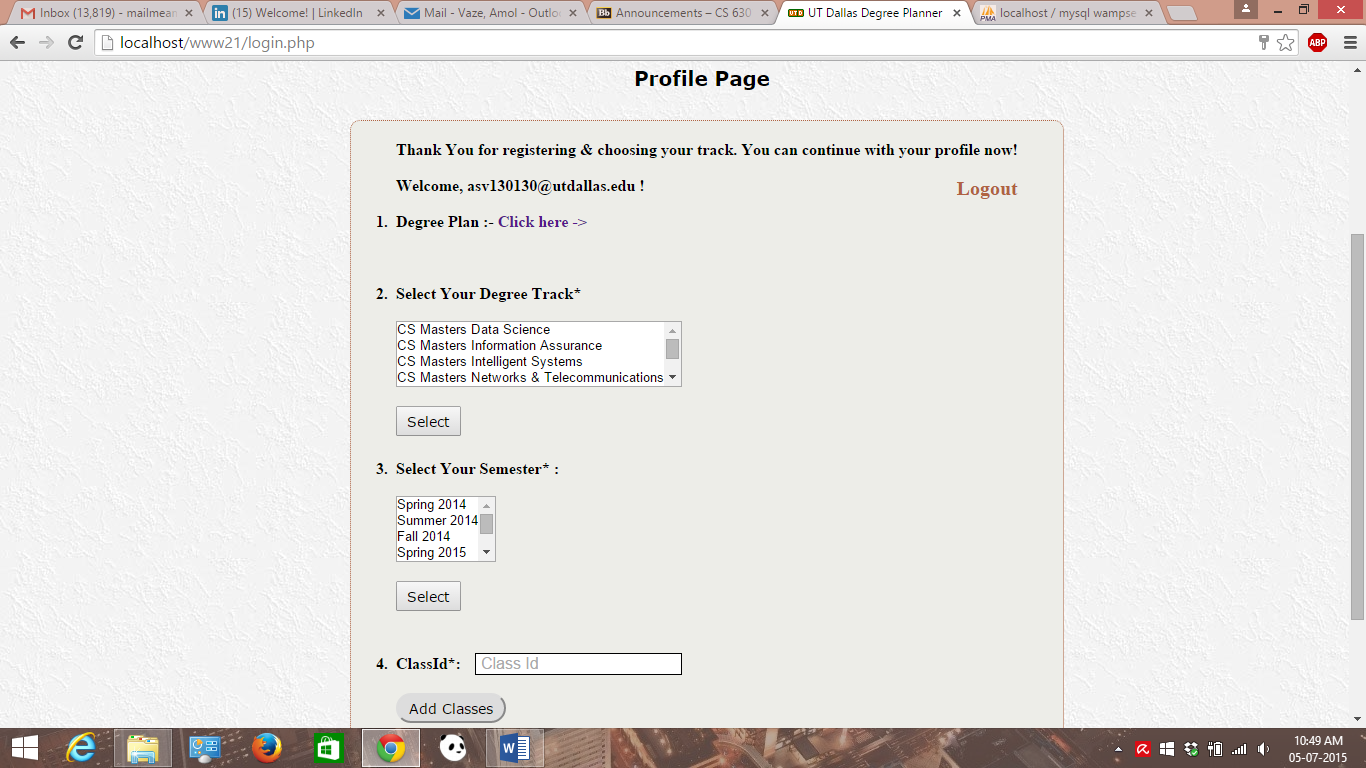
Registration Form validation is also provided using JavaScript when either fields left empty or if user enters some invalid email address.



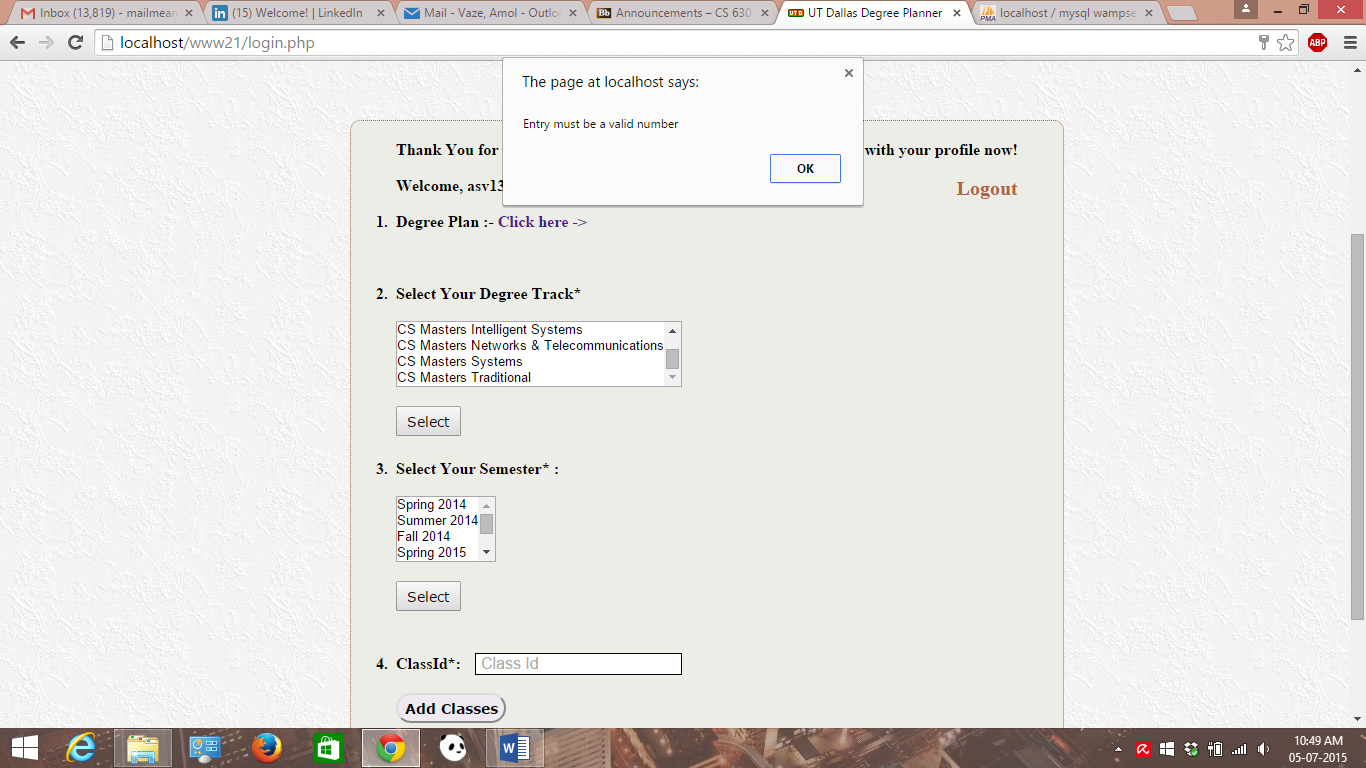
Also the validation has been provided when password and confirm password both do not match as shown below.



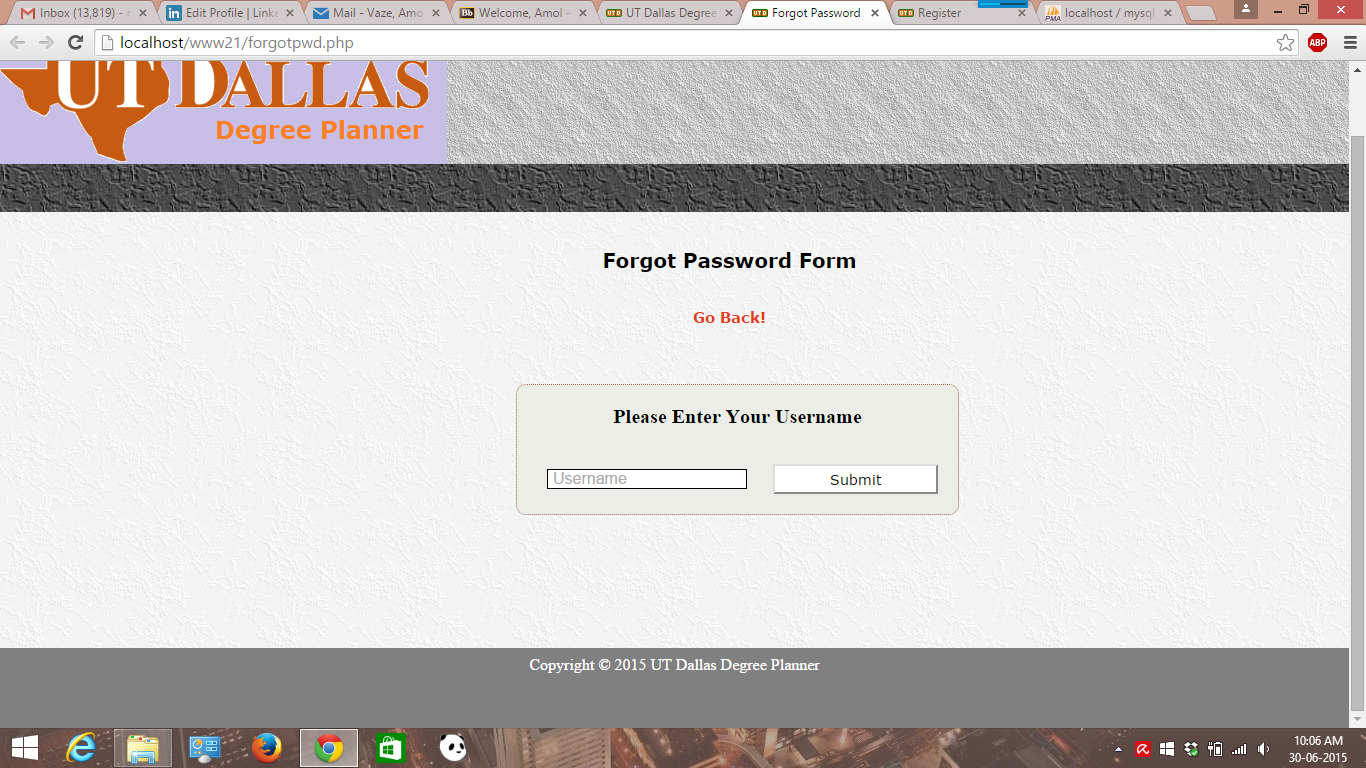
* **Login.php is** a web page which shows student user’s initial profile & it has the fields like Degree Plan out link which takes user to his/her degree plan page. Also, it has facility for track & semester selection. Form validation has been done for select & Add classes using JavaScript (client-side) which basically checks user is not leaving any field blank before proceeding and while entering classid, the entry should be numeric.

****

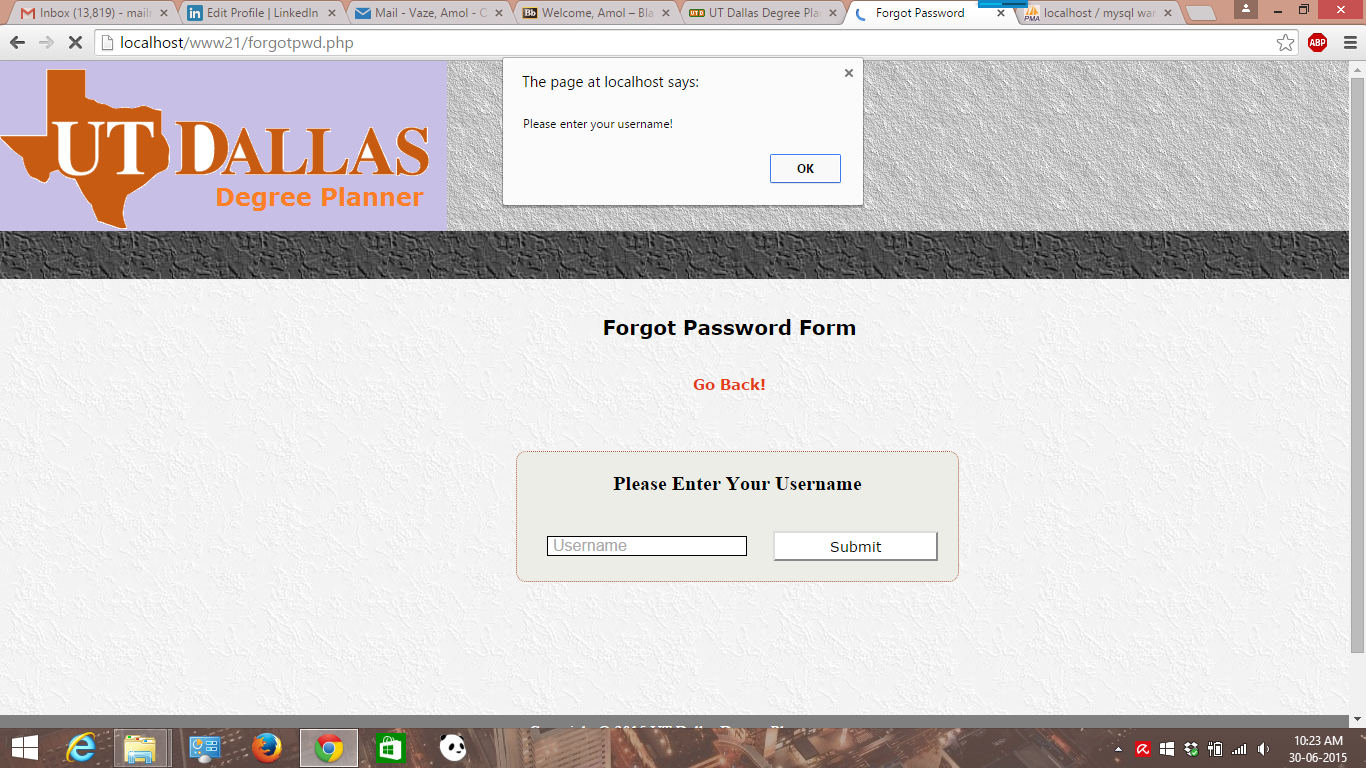
Profile page validation is done for fields as mentioned above. The corresponding screenshot for validation is attached below.

****

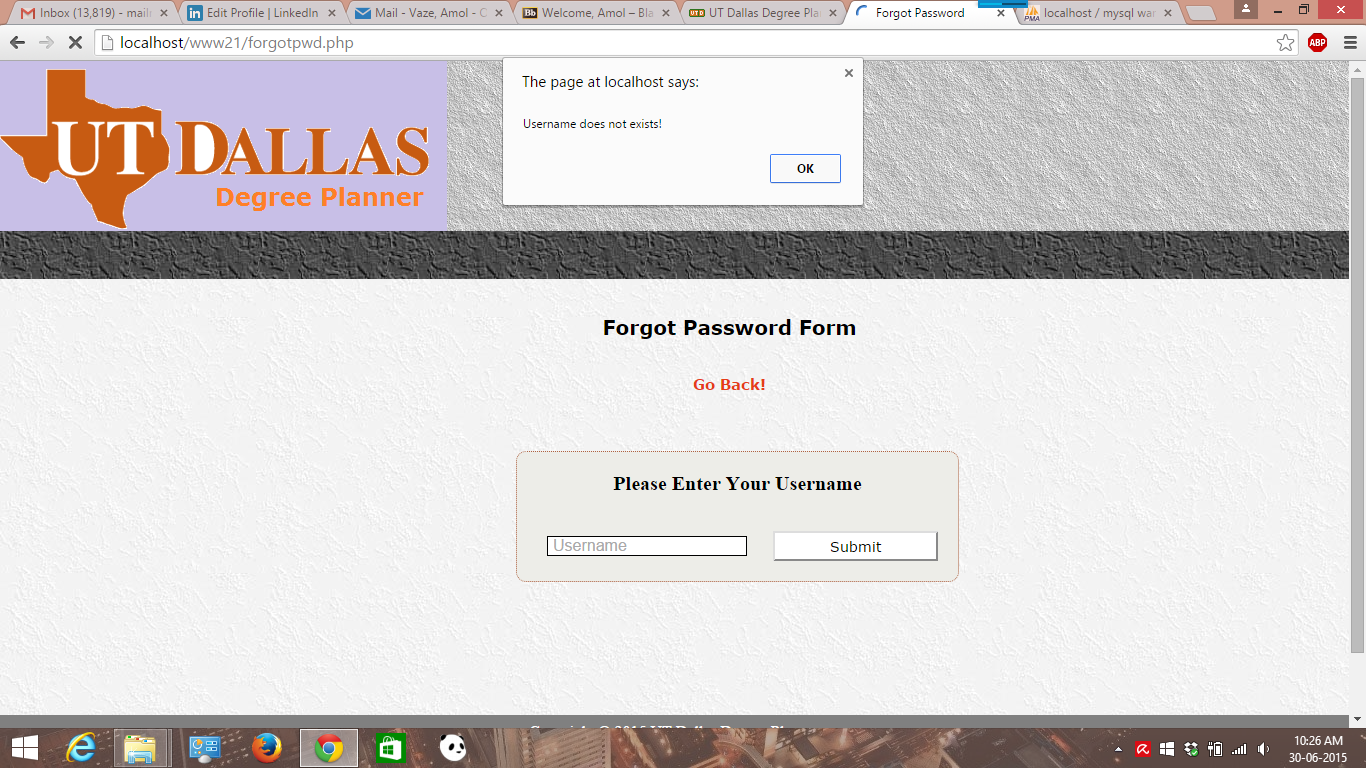
* **Logout.php** is responsible for logging the student user’s out of the system thereby destroying current user’s ongoing session. On clicking on this button, user will be redirected to home page of the system where he/she can log back into the system using credentials.
* **Forgot.php** isa web page which displays a simple form if user forgets his/her password and once accepting user’s name from the form, it will reset the corresponding password & resends that user a new credentials. It has a links to go back to home page. Using JavaScript(client side) validation is done.



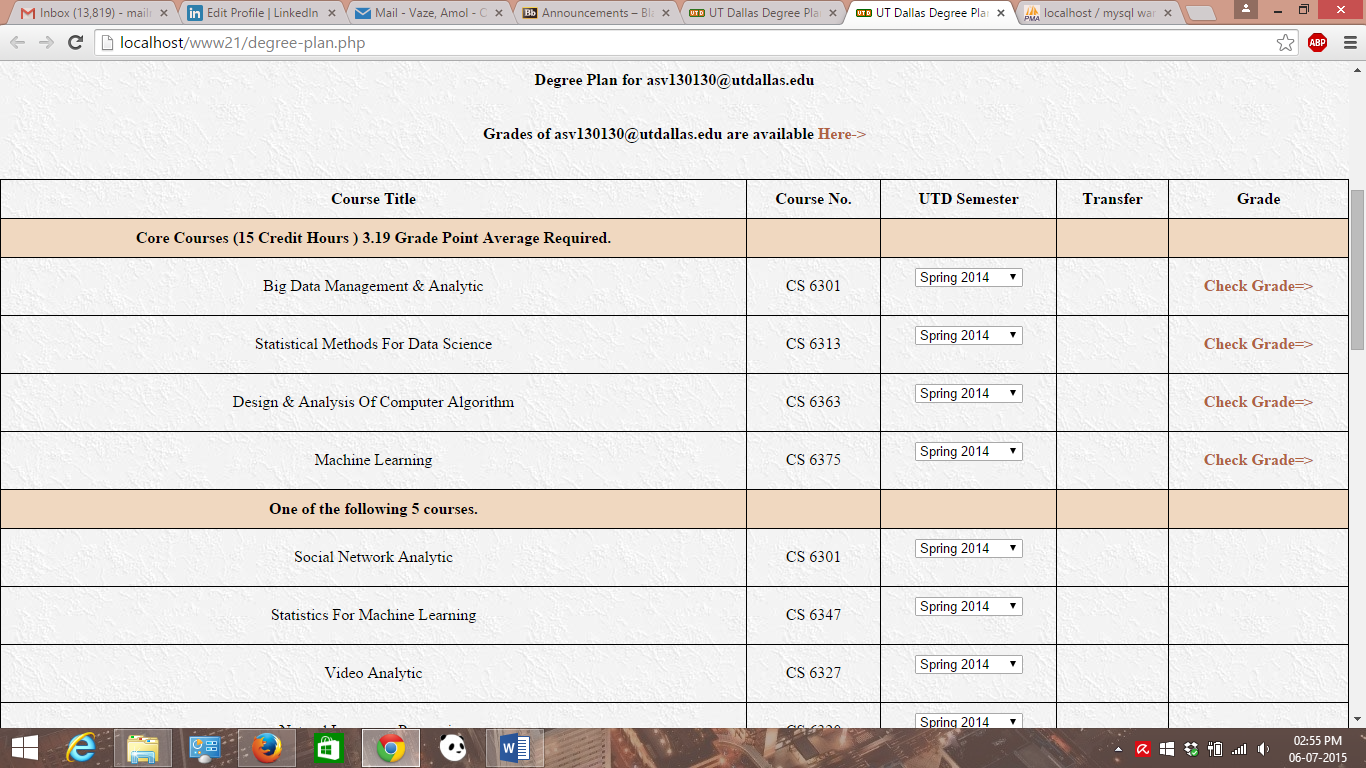
Forgot form validation is done here:-



Screenshot for validation part like user does not exists is shown in below screen shot:-

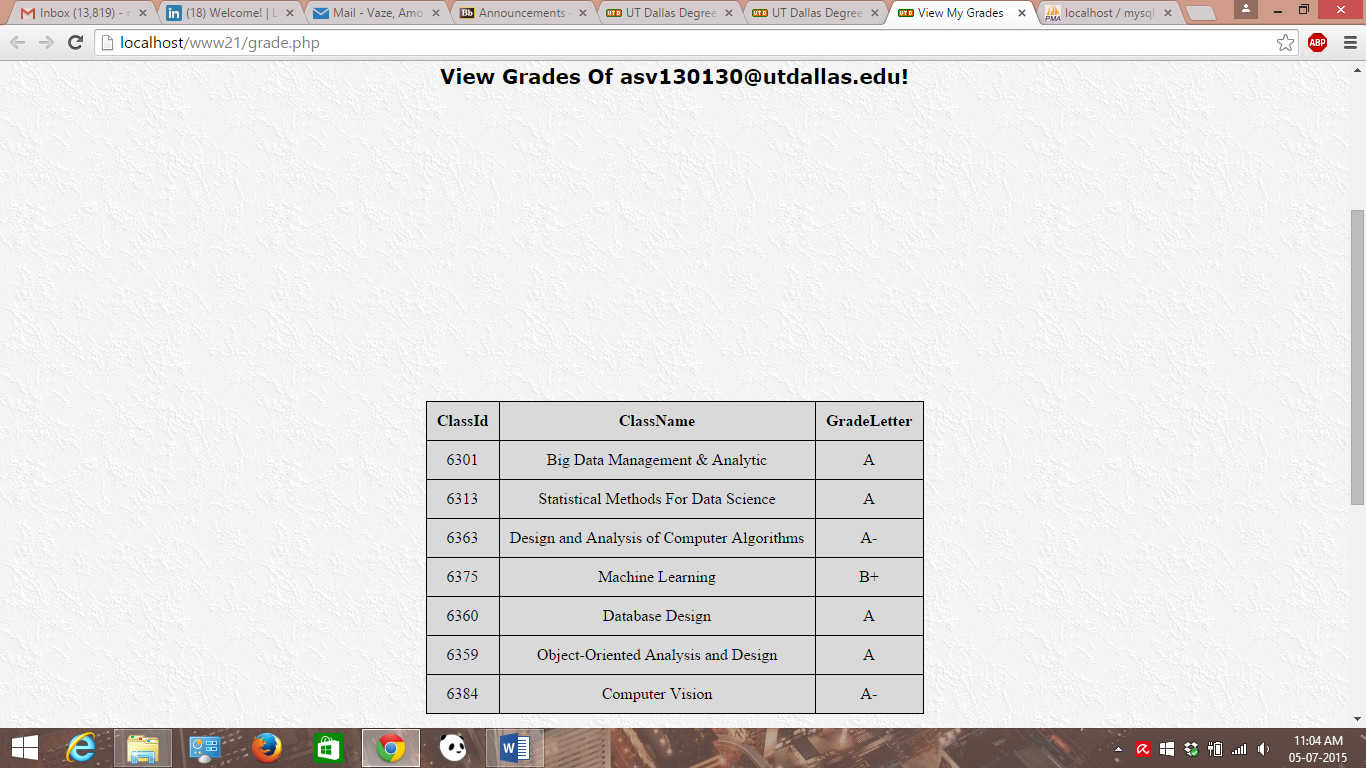


* **Degree-plan.php** is a web page where user should be able to see his/her degree plan table with all the core and elective courses he/she has registered for. User will be redirected to this page from profile page where it has a link.



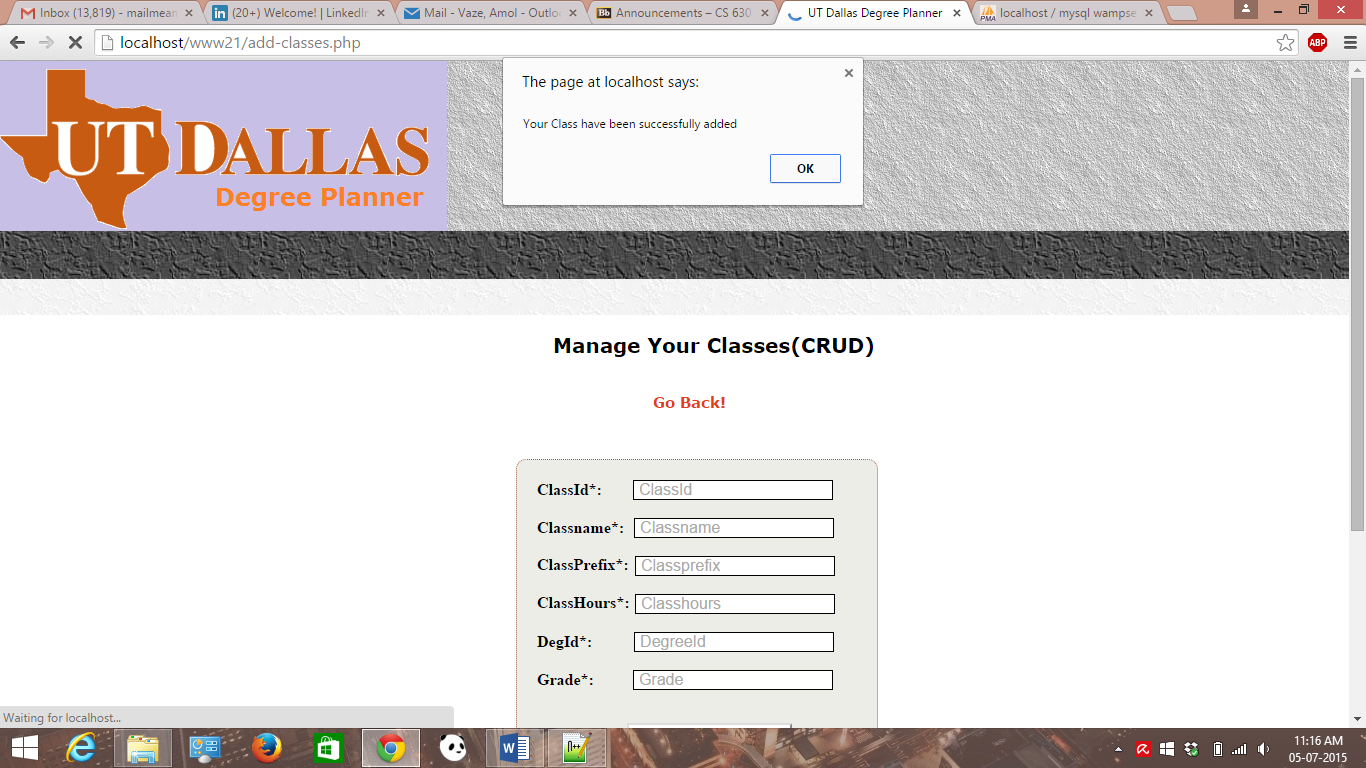
Degree-plan page has navigation link to go back to profile page in the navigation menu. Also degree plan page has a link where user should be able to see his/her grades.

* **Grade.php** is php file which fetches the content from database table & displays user’s grades for the courses he/she has taken. This table gets populated from the database tables at run time. This page also has the navigation link to go back. The screenshot for grades & courses is attached.

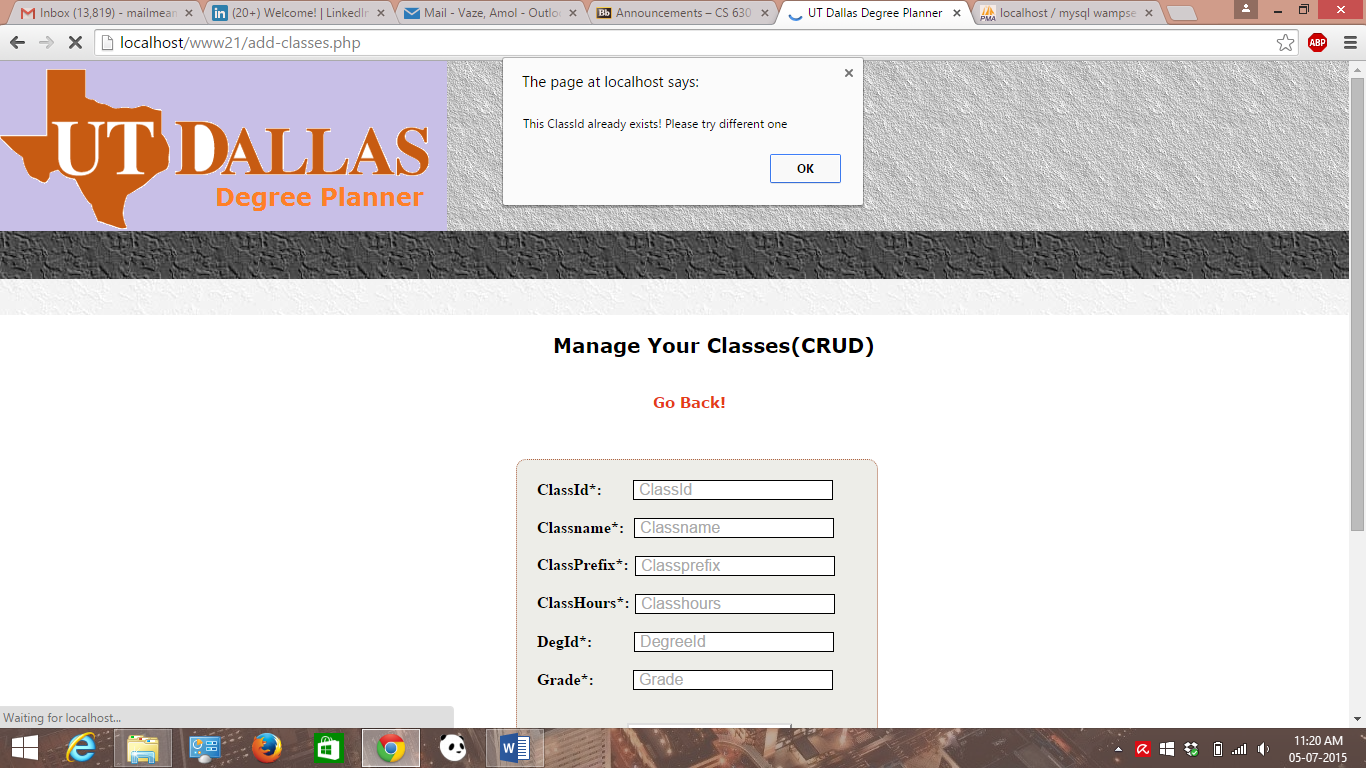


Class management screen shots (Add/Drop/Update) with JavaScript validation can be shown as follows:-

* **Add-classes.php** is a form which gives user to add new classes for which they want to register for. This simple forms accepts user’s details & inserts them into database table. JavaScript(client-side validation) is done.

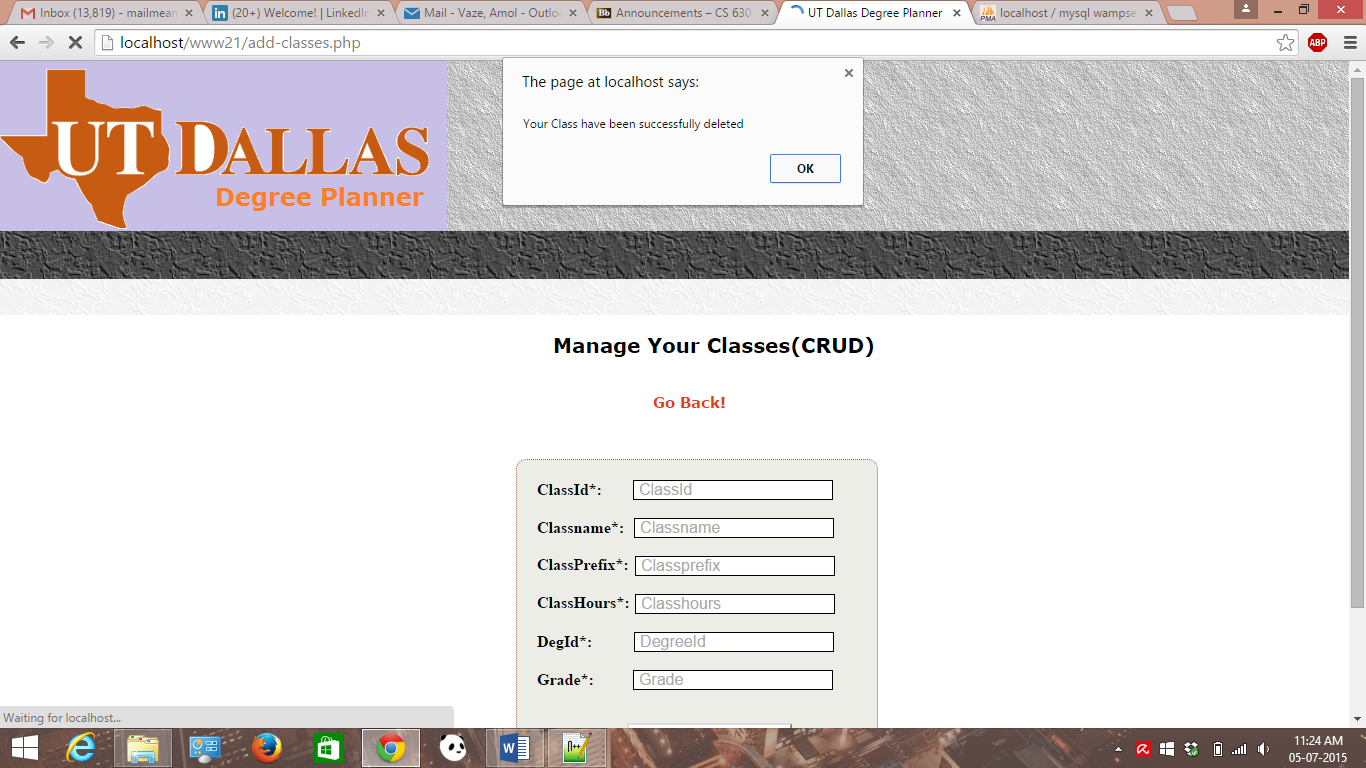
****

Add record/ class validation is done.



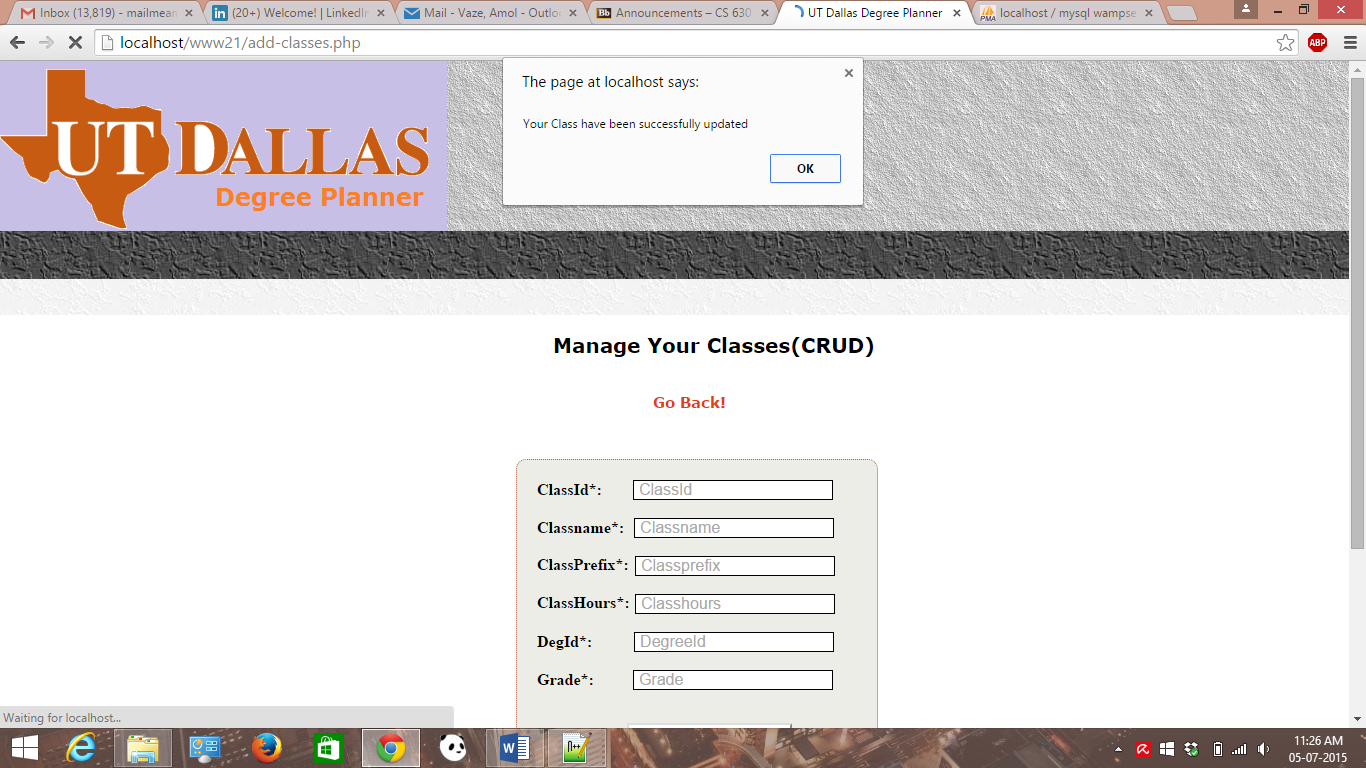
* **Drop-classes.php** is a form which gives user to drop (delete) existing classes for which they want to register for. This simple forms accepts user’s details & deletes them into database table. JavaScript(client-side validation) is done.

Screenshot is attached for drop/delete class below.



* **Update-classes.php** is form which gives user to update existing classes for which they want to register for. This simple forms accepts user’s details & updates them into database table. JavaScript(client-side validation) is done.

Screenshot is attached for update class.



**Database Tables For System & Sql codes for all tables**

The system has the schema for following database tables. A separate sql file/codes with all database tables is attached during submission.

* **Courses**- It has all the courses of UTD CS.
* **Degree Plan**- It has data related to various degree plans offered by UTD CS department.
* **Department**- This table lists all the available departments at UTD.
* **Grade** – This table has the list of grades for a particular student for particular class taken.
* **User** – This table has a list of all student users in the system.

Screenshots for all database tables as follows:-

User Table:-

Sql code for 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`)

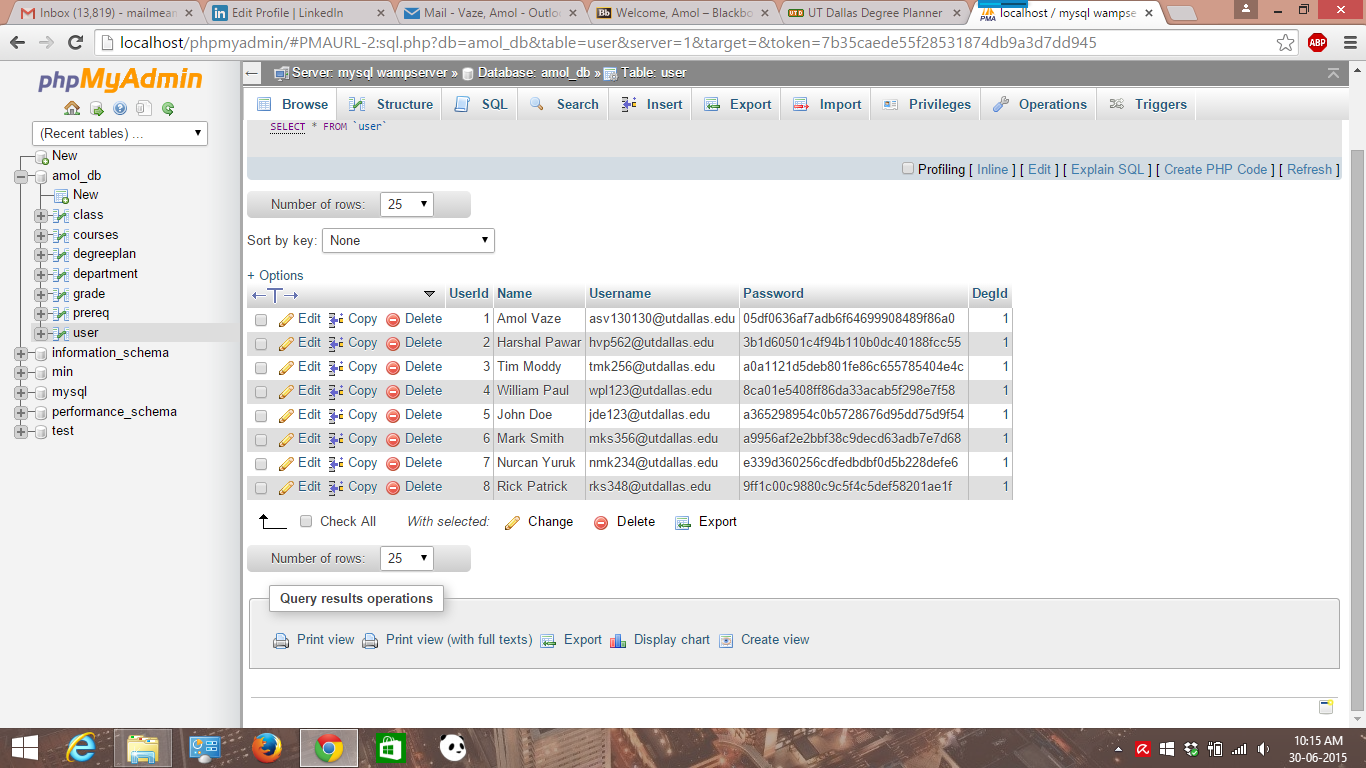
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=9 ;

-- Constraints for table `user`

--

ALTER TABLE `user`

ADD CONSTRAINT `user\_ibfk\_1` FOREIGN KEY (`DegId`) REFERENCES `degreeplan` (`DegId`);



Classes table:-

Sql Code for Classes 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`)

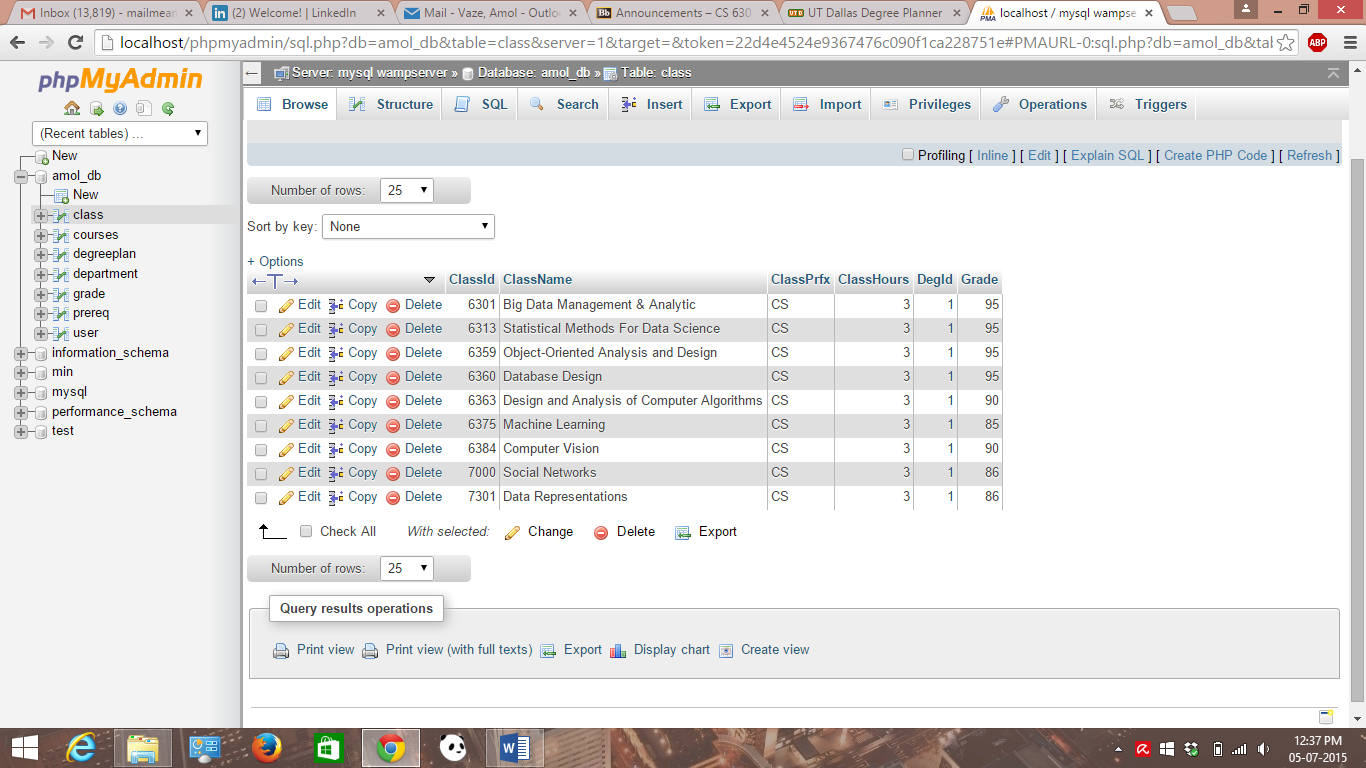
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- Constraints for table `class`

--

ALTER TABLE `class`

ADD CONSTRAINT `class\_ibfk\_1` FOREIGN KEY (`DegId`) REFERENCES `degreeplan` (`DegId`);



Courses table:-

Sql code for 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,

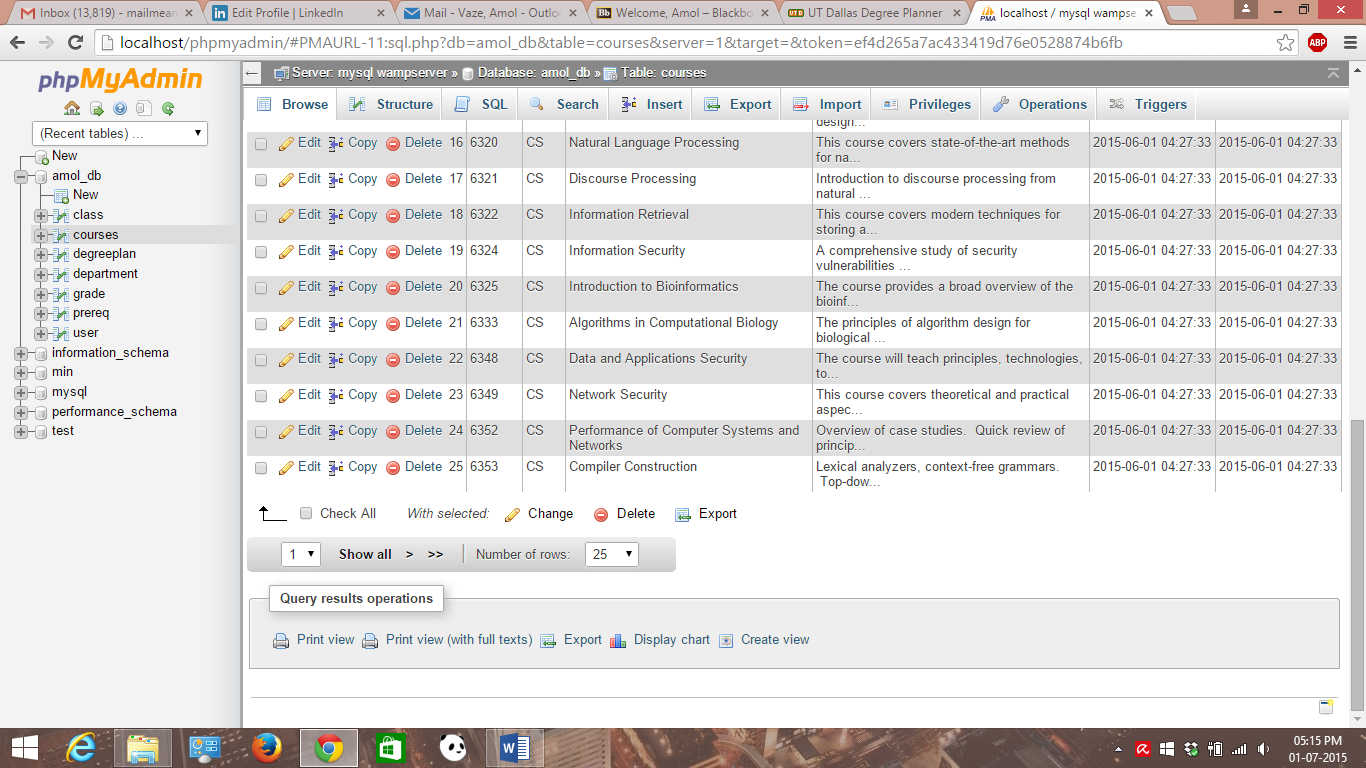
`description` text COLLATE utf8\_unicode\_ci,

`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 ;



Degree-plan table:-

Sql code for Degree-plan 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`)

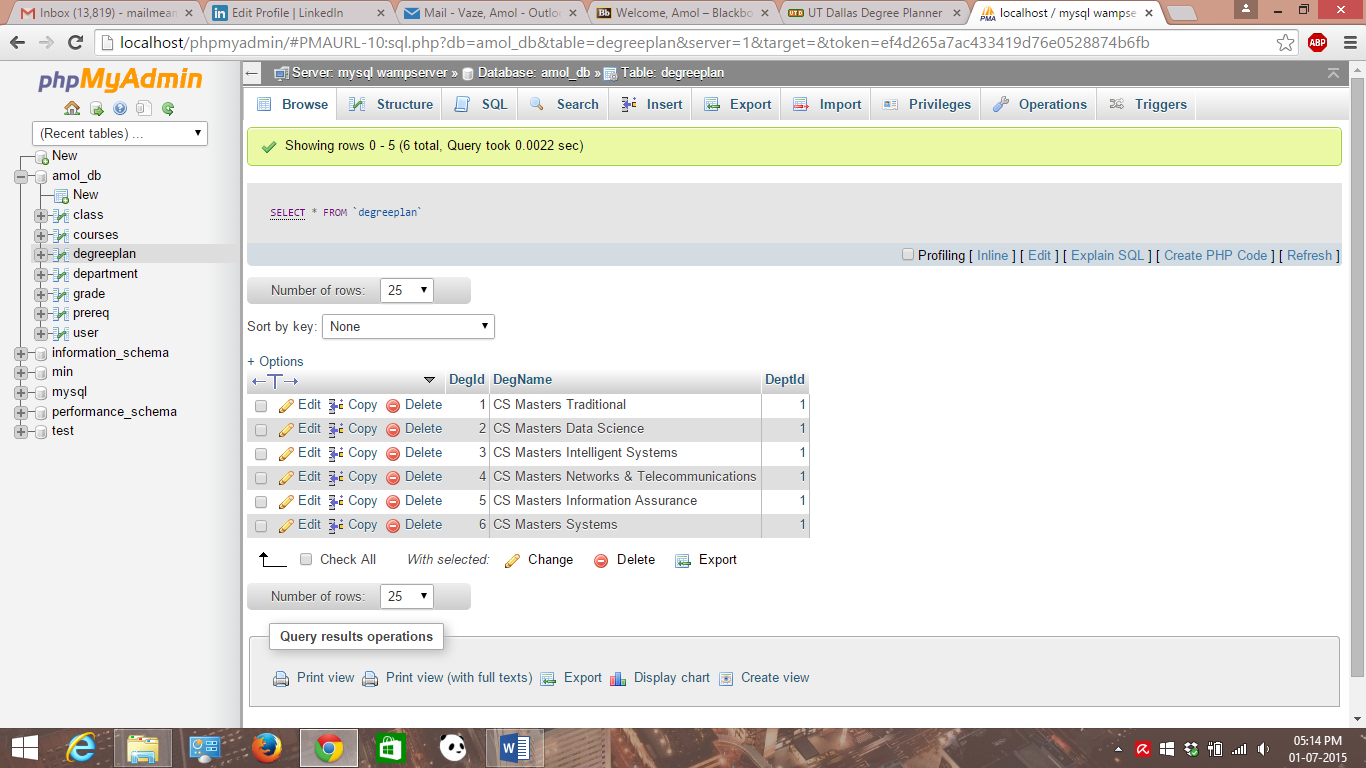
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=7 ;

-- Constraints for table `degreeplan`

--

ALTER TABLE `degreeplan`

ADD CONSTRAINT `degreeplan\_ibfk\_1` FOREIGN KEY (`DeptId`) REFERENCES `department` (`DeptId`);



Department table:-

Sql code for department table

CREATE TABLE IF NOT EXISTS `department` (

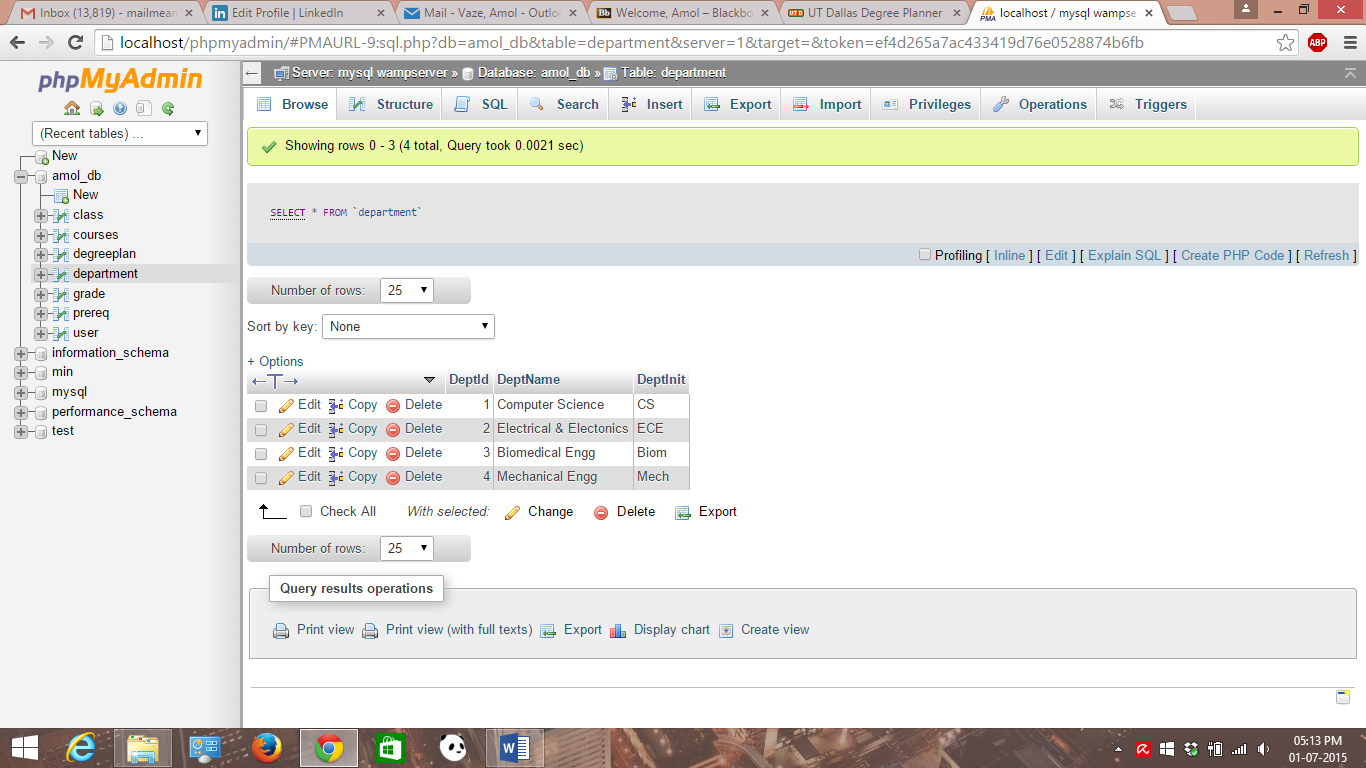
`DeptId` smallint(6) NOT NULL AUTO\_INCREMENT,

`DeptName` varchar(50) DEFAULT NULL,

`DeptInit` char(4) DEFAULT NULL,

PRIMARY KEY (`DeptId`)

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



Grade table:-

Sql code for 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,

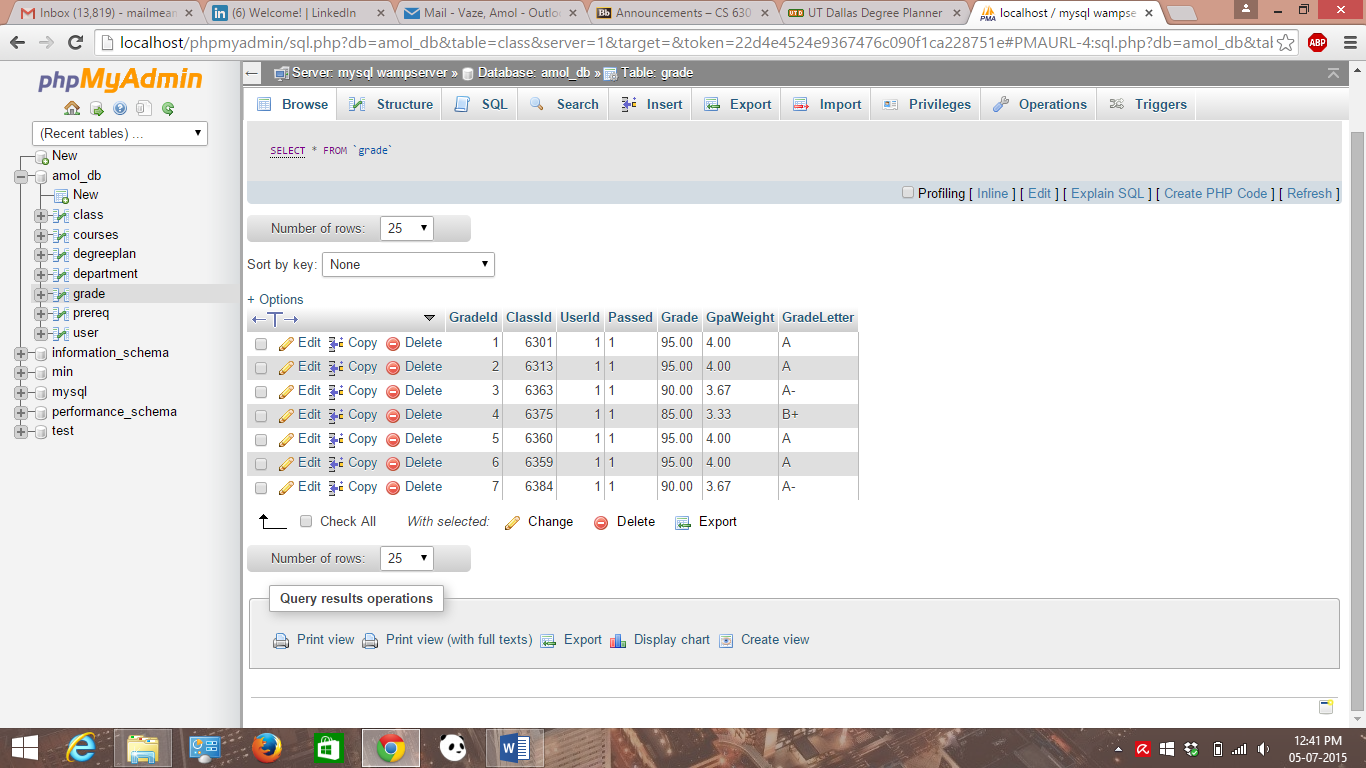
`GradeLetter` varchar(16) NOT NULL,

PRIMARY KEY (`GradeId`),

KEY `ClassId` (`ClassId`),

KEY `UserId` (`UserId`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=12 ;



**JavaScript Client-side Validation For System**

* The system needs to have client side validation which has been achieved using JavaScript.
* JavaScript validation has been done for pages like home page, profile page, main form page, forgot, register, main page classes’ pages.
* With the help of JavaScript functions, user must select required fields of the form before proceeding further.
* Also, functions take care of user’s input’s validity. It does check the need of numeric values accepted whenever needed. For example while entering classid on profile page form. Also check for radio button is done while selecting track on main form page.

**List of code (HTML, CSS, PHP)**

For list of code for all HTML,CSS & PHP, source code file have been attached along with submission-> www21 folder is there inside which all code files are stored.

​ **Web page (screenshot) with what php script is associated.**

Please note that for this point, it has already been discussed above screen shots with explanation (what is going on) step. It clearly depicts which PHP script is associated with which web page.

**Organization & easy to follow/read**

* This section describes the overall organisation of the system in which a system starts from the home page which is a login page.
* After entering login details & after successful input validation, user will be allowed to proceed with the next main form page where a particular track selection will be done.
* After track selection & login, user will be taken to initial profile page where user can have option to log out of the system at RHS top corner.
* This profile page gives user to add new classes at last step of the profile page & also user should be able to view grades on the link provided for the user’s degree plan at very first step.
* Necessary input validation is done for example for numeric entry for class id and please select to proceed for the select button control.
* Once user is taken to manage classes page (CRUD), user should be able to add, drop and update classes/ grades through the form provided. This page has navigation to go back to the profile page.
* Also, the needed input validation is done for manage classes page.
* Once user clicks on the link to check out degree plan, then a corresponding degree plan web page is shown to the user with all the core and elective courses he has been taken. This page has a link with the help of which user should be able to see his grades. Also, there is a link “check grades” for viewing grades of the course in degree plan table.
* Once user clicks on the view grades link, user should be able to see the grades for all the registered courses. This page also has the navigation to go back.
* Finally on the profile page when user logs out by clicking on the log out link & then user will be redirected to the home page of the system.

**P.S [The system does not use separate table for login, logout and register but, it has a only one table named User that is performing the function of these 3 tables.]**

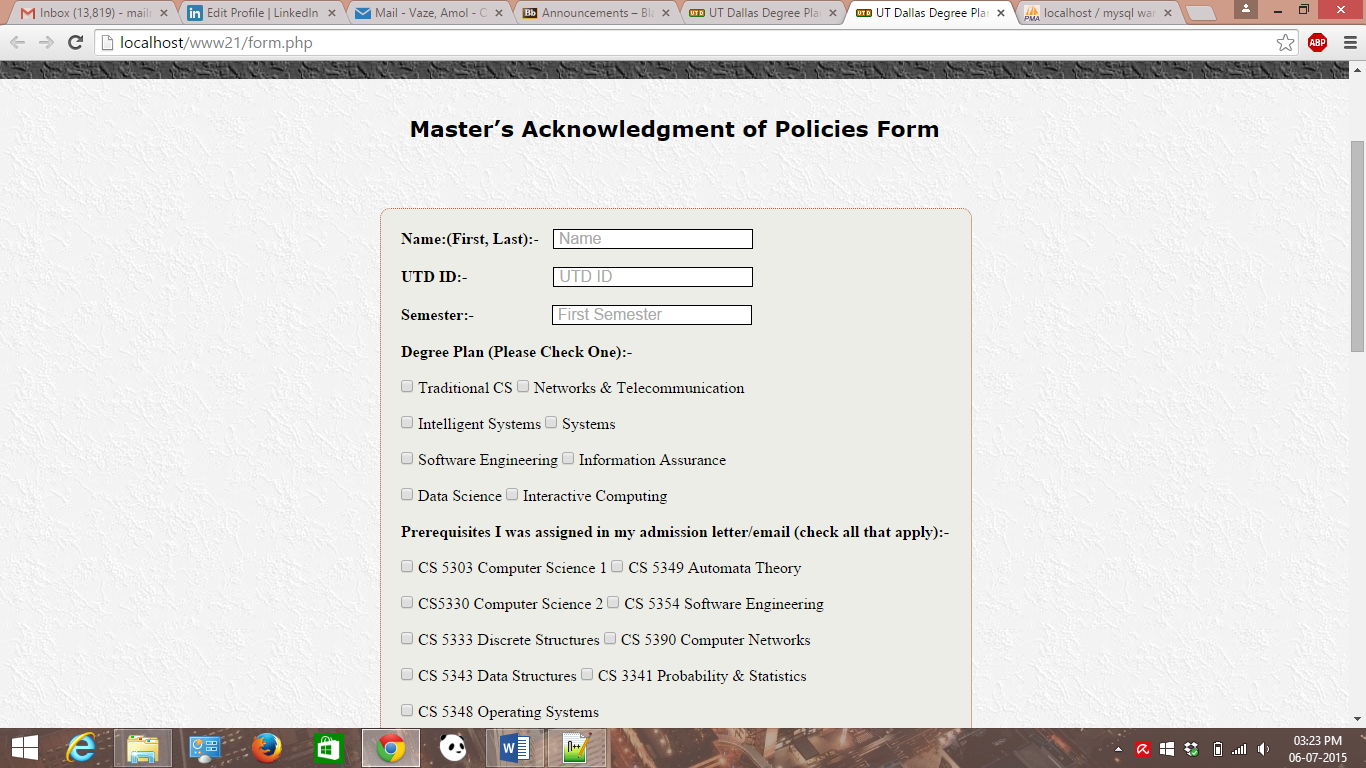
**How to relate a HTML web page with form variable names to/from DB tables**

* In order to relate HTML web page with form variable names to/from database table consider an example of Register Form where new user is doing registration by filling out the form details.
* For example, in case of registration form, we are having different HTML elements like Name, Username, Password & finally the Register Button.
* Hence, on the same page of Register.php there is database connectivity is done via PHP code.
* In this code, we know that each HTML form element can be easily identified with the help of Id attribute assigned to it.
* Hence, using Isset function of PHP, first we are checking whether a particular form element is set or not and it has to be non-empty.
* Once this condition has been fulfilled, then PHP variable is assigned with that value (The POST method is used) that will communicate with database table. This means that this variable can be used in SQL query statement which plays role in the database interaction.
* For example, $sql="INSERT INTO user (userid, name,username,password,degId) VALUES('','$name','$user','$pass','$degId')" is a SQL query and it has all the required variables which have been set & those corresponds to the HTML form elements.
* In above example, $name, $user, $pass etc are the variable names used to/from connection with the database table; in this example User table (As mentioned earlier)
* Once the user has entered all the details successfully required by the register form, then a new record will be inserted into the user table of the database.

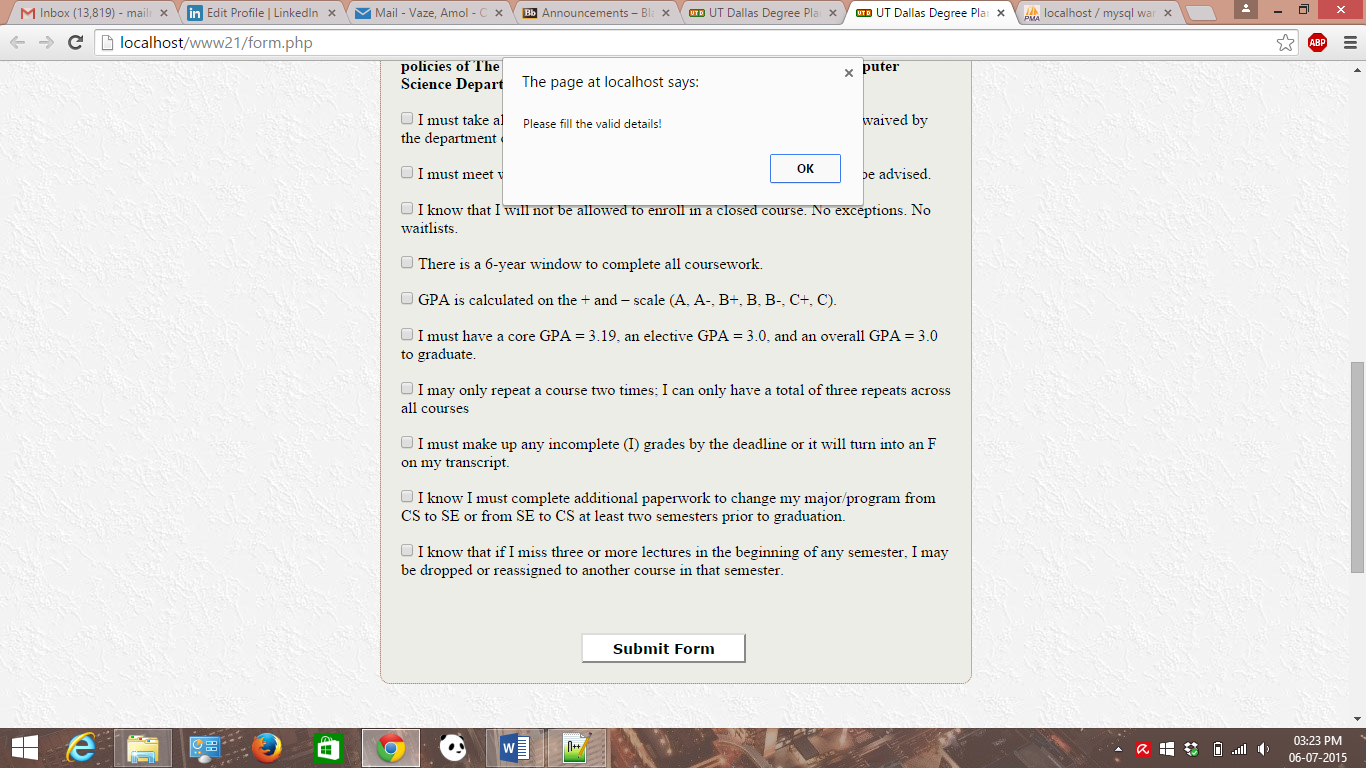
**Navigation or main flow of the process**

* This section gives a brief overview of the navigation or main flow of the process where main flow of the system starts from the home page.
* As the user logs in into the system b successfully registering, user’s session will be started till the user logs out of the system back.
* Profile page has the navigation menu of degree requirement where user should be able to see the general degree requirement. This degree requirement page has the navigation link to go back again.
* Profile page also has the similar menu of courses where user should be able to see the various courses list along with the course description. This courses page has the navigation link to go back again.
* Again, profile page has the link for viewing degree plan for the user which in turn has the navigation to check grades for the courses taken.
* Both pages have links for to and fro navigation.
* Finally, when user logs out of the system, user will be navigated to home page where user should be able to log back into the system.

**P.S [Design Decision: - I have created Acknowledgement form and I have provided JavaScript Validation for the same. But I am planning to use it for the next assignment; for implementation of multiple tracks.]**

****

Now, acknowledgement form has provided with validation usng JavaScript & its screenshot is attached below.

****

**Conclusion**

To conclude, this document explains the step wise flow of the entire system for the degree planner in detail.