Student Academic Database System

Rashmi Lengade

UTD ID: 2021147362

**NARRATIVE**

This Student Academic Database System is a web-based application developed to make an easier and more convenient form of class registration process, a hassle through which students go every semester.

As it stands, here at University of Texas at Dallas, in order to officially register for classes, each student must reach the advisors office and fill out a course registration form manually and have the form approved from the advisors. In order to change anything about one’s current schedule, such as dropping or adding a class, changing grade option, or showing official permission from a professor, the student must go through the same tedious process.

Student Academic Database System (SADS) attempts to eliminate these hassles by providing service to students through the internet. SADS provides a way to search for classes without having to open a course catalog, a way to register for classes and view various possible schedules, and finally, officially register for the chosen classes. All this can be done in the at the students’ convenience and eliminate the time reach the graduate advisor office for the registration of courses.

The University of Texas at Dallas (UTD) has numerous departments each of which offers various courses. Every department at the end of each semester has to collect feedback from professors for the courses they taught. The feedback is then used by department to identify the problems areas and provide solutions which would help the future students who would be undertaking the course to get a better understanding of that course.

The course lookup is available for the student to look through the available courses before they register. The course look up has the details about the courses offered life the name of the course, the class timings, room where class will be taken and so on.

The new proposed system would make the registration of courses online and hence would help in overcoming the drawbacks of the current manual system. The registration details would be stored in a centralized repository and hence accessing registration details would be easy and efficient. The system would be used by instructors to fill and submit the grades of the students online and also view the details of the students registered for the course offered by the instructor.

**Functional Requirements:**

The users who would be using the system are students and the instructors. The functional requirements provided by the system for the students and the instructors are given as follows:

**Student:**

1. The student logs into the system using his/her NetID and password.
2. On successful login the student can search for available courses.

* By Semester :

The student can select a semester from the list provided and then view the list of courses offered in that semester.

* By Course Name:

The student can enter a course name and then view the list of courses offered in that semester.

* By Course Number:

The student can enter a course number and then view the list of courses offered in that semester.

* By Instructor Name:

The student can enter the instructor name and then view the list of courses offered in that semester.

* By Class time:

The student can enter a course name and then view the list of courses offered in that semester.

1. The student can register for a course by entering the course details.
2. The student can drop a course that he/she has registered.

**Instructor:**

1. The professor can view the student registered in the course which the professor is teaching.
2. The professor can assign grades to the students registered in the course which the professor is teaching.

**Data Requirements:**

1. The SSN uniquely identifies each tuple of the Instructor table and hence should be unique and cannot be NULL.
2. The student ID uniquely identifies each tuple of the student table and hence should be unique and cannot be NULL.
3. The Course Number of each course has a 4 digit number. The Course Number uniquely identifies each tuple of the Course table and hence cannot be NULL.
4. The section ID uniquely identifies each tuple of the section table and hence should be unique and cannot be NULL.
5. The NetID of Instructor is a combination of initials of the instructor's name and a 6 digit random number.

**Mapping from ER Model to Relational Model:**

1. COLLEGE(COLLEGEID,COLLEGENAME,ADDRESS)

COLLEGEID-Primary Key

1. STUDENT(*SID*,FNAME,LNAME,STREET, CITY,STATE,ZIPCODE,PHONENO,SEX,BDATE,NETID,STUDENTTYPE,MAJOR,DID,ASSN)

SID – Primary Key

DID – Foreign Key (DEPARTMENT)

1. DEPARTMENT (DID,DNAME,DLOCATION,DSSN,COLLEGEID)

DID – Primary Key

SSN – Foreign Key (STAFF)

COLLEGEID –Foreign Key(COLLEGE)

1. STAFF(SSN,FNAME,LNAME,SEX,STREET,CITY,STATE,ZIPCODE,PHONENO,OFFICEHRS,OFFICELOCATION,NETID,SALARY,RANK,DID,PASSWORD)

SSN– Primary Key

DID-Foreign Key

1. COURSE (COURSEID,COURSENAME,CDESCRIPTION,TYPE,LEVEL,DID)

COURSEID – Primary Key

DID – Foreign Key (DEPARTMENT)

1. SECTION (SECTIONNO,SECTIONID,SEMESTER,YEAR,LECTURE\_DAY,START\_TIME,END\_TIME,STATUS,LEVEL,CREDIT\_HOURS,CAPACITY,LOCATION,INSTRUCTORSSN,COURSEID)

SECTIONID – Primary Key

COURSEID,INSTRUCTORSSN – Foreign Key (COURSE, STAFF)

1. GRADE\_REPORT (SID,SECTIONID,NUMBER\_GRADE,GRADE,GUIDE\_SSN)

SID,SECTIONID – Primary Key

GUIDE\_SSN – Foreign Key (STAFF)

1. PREREQUISITE(MAINCOURSEID,PREREQCOURSEID)

MAINCOURSEID,PREREQCOURSEID- Primary Key

MAINCOURSEID,PREREQCOURSEID-Foreign Key

1. STUDENT\_RA(SID,OFFICELOCATION,GUIDE\_SSN)

SID-Primary Key

SID-Foreign Key

1. STUDENT\_TA(SID,SECTIONID)

SID-Primary Key

SID,SECTIONID-Foreign Key(STUDENT,SECTION)

1. PROJECT(PROJECTNAME,SID,PROJECTDESC)

PROJECTNAME,SID-Primary Key

SID-Foreign Key(STUDENT)

1. THESIS(THESISID,THESISNAME,THESISDESC,SSN,SID)

THESISID-Primary Key

SID,SSN-Foreign Key(STUDENT,STAFF)

**Tables in 3NF:**

The final tables in 3NF can be given as -

**COLLEGE**(COLLEGEID,COLLEGENAME,ADDRESS)

**Functional Dependencies:**

COLLEGEID🡪COLLEGENAME,ADDRESS

**DEPARTMENT** (DID,DNAME,DLOCATION,DSSN,COLLEGEID)

**Functional Dependencies:**

DID🡪 DNAME,DLOCATION,DSSN,COLLEGEID

**STUDENT**(SID,FNAME,LNAME,STREET,CITY,STATE,ZIPCODE,PHONENO,SEX,BDATE,NETID,STUDENTTYPE,MAJOR,DID,ASSN)

**Functional Dependencies:**

SID🡪FAME,LNAME,STREET,CITY,STATE,ZIPCODE,PHONENO,SEX,BDATE,STUDENTTYPE,MAJOR,DID,ASSN

 NETID🡪FAME,LNAME,STREET,CITY,STATE,ZIPCODE,PHONENO,SEX,BDATE,STUDENTTYPE,MAJOR,DID,ASSN

**STAFF**(SSN,FNAME,LNAME,SEX,STREET,CITY,STATE,ZIPCODE,PHONENO,OFFICEHRS,OFFICELOCATION,NETID,SALARY,RANK,DID,PASSWORD)

**Functional Dependencies:**

SSN🡪FNAME,LNAME,SEX,STREET,CITY,STATE,ZIPCODE,PHONENO,OFFICEHRS,OFFICELOCATION,SALARY,RANK,DID,PASSWORD

NETID🡪FNAME,LNAME,SEX,STREET,CITY,STATE,ZIPCODE,PHONENO,OFFICEHRS,OFFICELOCATION,SALARY,RANK,DID,PASSWORD

**COURSE** (COURSEID,COURSENAME,CDESCRIPTION,TYPE,LEVEL,DID)

**Functional Dependencies:**

COURSEID🡪 COURSENAME,CDESCRIPTION,TYPE,LEVEL,DID

**SECTION**(SECTIONNO,SECTIONID,SEMESTER,YEAR,LECTURE\_DAY,START\_TIME,END\_TIME,STATUS,LEVEL,CREDIT\_HOURS,CAPACITY,LOCATION,INSTRUCTORSSN,COURSEID)

**Functional Dependencies:**

SECTIONID🡪SECTIONNO,SEMESTER,YEAR,LECTURE\_DAY,START\_TIME,END\_TIME,STATUS,LEVEL,CREDIT\_HOURS,CAPACITY,LOCATION,INSTRUCTORSSN,COURSEID

**GRADE\_REPORT** (SID,SECTIONID,NUMBER\_GRADE,GRADE,GUIDE\_SSN)

**Functional Dependencies:**

SID,SECTIONID🡪 NUMBER\_GRADE,GRADE,GUIDE\_SSN

**STUDENT\_RA**(SID,OFFICELOCATION,GUIDE\_SSN)

**Functional Dependencies:**

SID🡪 OFFICELOCATION,GUIDE\_SSN

**PROJECT**(PROJECTNAME,SID,PROJECTDESC)

**Functional Dependencies:**

PROJECTNAME,SID🡪PROJECTDESC

**THESIS**(THESISID,THESISNAME,THESISDESC,SSN,SID)

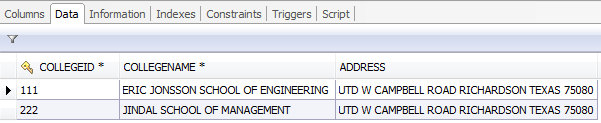
**Functional Dependencies:**

THESISID🡪 THESISNAME,THESISDESC,SSN,SID

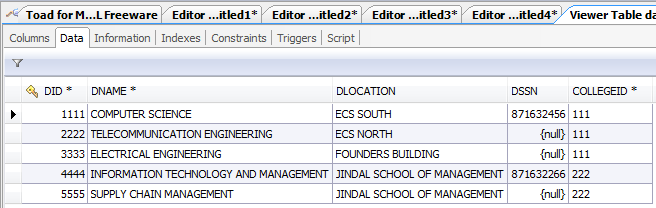
**All the above FDs are enforced.**

**Data stored in all tables:**

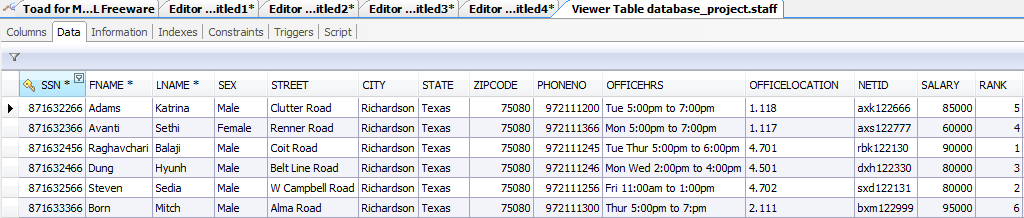
1.**COLLEGE:**

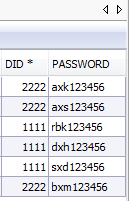
****

2. **DEPARTMENT:**

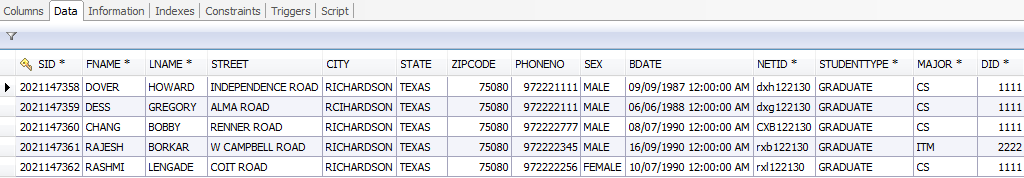


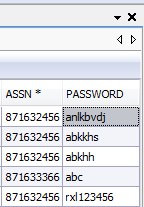
**3.STAFF:**

****

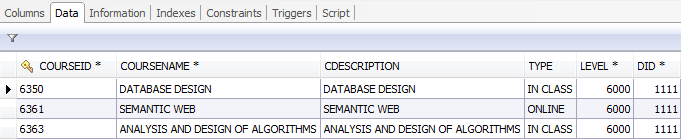
****

**4. STUDENT:**

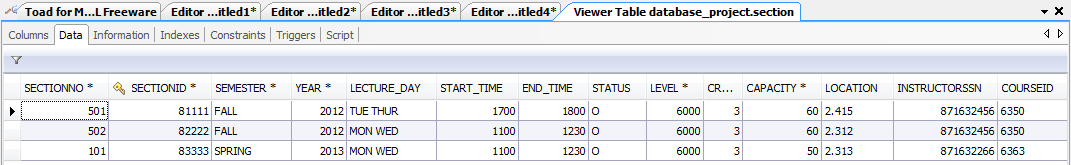




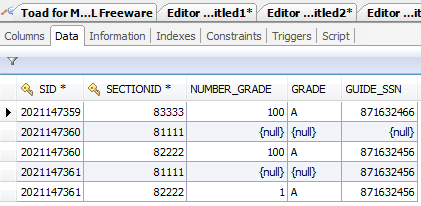
**5.COURSE:**



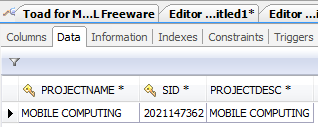
**6.SECTION:**

****

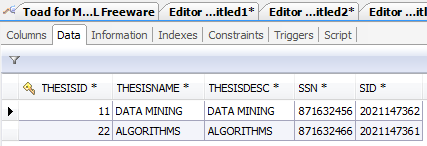
**7.GRADE REPORT:**

****

**8.PROJECT:**

****

**9.THESIS:**

****

**SQL Commands used to create tables:**

**CREATE TABLE `college` (**

**`COLLEGEID` varchar(10) NOT NULL,**

**`COLLEGENAME` varchar(50) NOT NULL,**

**`ADDRESS` varchar(50) DEFAULT NULL,**

**PRIMARY KEY (`COLLEGEID`)**

**)**

**CREATE TABLE `department` (**

**`DID` int(11) NOT NULL,**

**`DNAME` varchar(50) NOT NULL,**

**`DLOCATION` varchar(60) DEFAULT NULL,**

**`DSSN` int(9) DEFAULT NULL,**

**`COLLEGEID` varchar(10) NOT NULL,**

**PRIMARY KEY (`DID`),**

**KEY `FK\_College\_Id` (`COLLEGEID`),**

**KEY `SSN\_FK` (`DSSN`),**

**CONSTRAINT `FK\_College\_Id` FOREIGN KEY (`COLLEGEID`) REFERENCES `college` (`COLLEGEID`),**

**CONSTRAINT `SSN\_FK` FOREIGN KEY (`DSSN`) REFERENCES `staff` (`SSN`)**

**)**

**CREATE TABLE `staff` (**

**`SSN` int(9) NOT NULL,**

**`FNAME` varchar(20) NOT NULL,**

**`LNAME` varchar(20) NOT NULL,**

**`SEX` varchar(20) DEFAULT NULL,**

**`STREET` varchar(50) DEFAULT NULL,**

**`CITY` varchar(50) DEFAULT NULL,**

**`STATE` varchar(20) DEFAULT NULL,**

**`ZIPCODE` int(5) DEFAULT NULL,**

**`PHONENO` int(10) DEFAULT NULL,**

**`OFFICEHRS` varchar(50) DEFAULT NULL,**

**`OFFICELOCATION` varchar(10) DEFAULT NULL,**

**`NETID` varchar(9) DEFAULT NULL,**

**`SALARY` int(11) DEFAULT NULL,**

**`RANK` int(11) DEFAULT NULL,**

**`DID` int(11) NOT NULL,**

**`PASSWORD` varchar(10) DEFAULT NULL,**

**PRIMARY KEY (`SSN`),**

**KEY `FORIEGN\_KEY` (`DID`),**

**CONSTRAINT `FORIEGN\_KEY` FOREIGN KEY (`DID`) REFERENCES `department` (`DID`)**

**)**

**CREATE TABLE `staff` (**

**`SSN` int(9) NOT NULL,**

**`FNAME` varchar(20) NOT NULL,**

**`LNAME` varchar(20) NOT NULL,**

**`SEX` varchar(20) DEFAULT NULL,**

**`STREET` varchar(50) DEFAULT NULL,**

**`CITY` varchar(50) DEFAULT NULL,**

**`STATE` varchar(20) DEFAULT NULL,**

**`ZIPCODE` int(5) DEFAULT NULL,**

**`PHONENO` int(10) DEFAULT NULL,**

**`OFFICEHRS` varchar(50) DEFAULT NULL,**

**`OFFICELOCATION` varchar(10) DEFAULT NULL,**

**`NETID` varchar(9) DEFAULT NULL,**

**`SALARY` int(11) DEFAULT NULL,**

**`RANK` int(11) DEFAULT NULL,**

**`DID` int(11) NOT NULL,**

**`PASSWORD` varchar(10) DEFAULT NULL,**

**PRIMARY KEY (`SSN`),**

**KEY `FORIEGN\_KEY` (`DID`),**

**CONSTRAINT `FORIEGN\_KEY` FOREIGN KEY (`DID`) REFERENCES `department` (`DID`)**

**)**

**CREATE TABLE `course` (**

**`COURSEID` varchar(9) NOT NULL,**

**`COURSENAME` varchar(50) NOT NULL,**

**`CDESCRIPTION` varchar(40) DEFAULT NULL,**

**`TYPE` varchar(10) DEFAULT NULL,**

**`LEVEL` int(11) NOT NULL,**

**`DID` int(11) NOT NULL,**

**PRIMARY KEY (`COURSEID`),**

**KEY `DID\_FK` (`DID`),**

**CONSTRAINT `DID\_FK` FOREIGN KEY (`DID`) REFERENCES `department` (`DID`)**

**)**

**CREATE TABLE `section` (**

**`SECTIONNO` int(11) NOT NULL,**

**`SECTIONID` int(11) NOT NULL,**

**`SEMESTER` varchar(6) NOT NULL,**

**`YEAR` int(4) NOT NULL,**

**`LECTURE\_DAY` varchar(20) DEFAULT NULL,**

**`START\_TIME` int(4) DEFAULT NULL,**

**`END\_TIME` int(4) DEFAULT NULL,**

**`STATUS` char(1) DEFAULT NULL,**

**`LEVEL` int(11) NOT NULL,**

**`CREDIT\_HOURS` int(2) NOT NULL,**

**`CAPACITY` int(3) NOT NULL,**

**`LOCATION` varchar(20) DEFAULT NULL,**

**`INSTRUCTORSSN` int(11) DEFAULT NULL,**

**`COURSEID` varchar(9) DEFAULT NULL,**

**PRIMARY KEY (`SECTIONID`),**

**KEY `section\_ibfk\_1` (`INSTRUCTORSSN`),**

**KEY `section\_ibfk\_2` (`COURSEID`),**

**CONSTRAINT `section\_ibfk\_1` FOREIGN KEY (`INSTRUCTORSSN`) REFERENCES `staff` (`SSN`),**

**CONSTRAINT `section\_ibfk\_2` FOREIGN KEY (`COURSEID`) REFERENCES `course` (`COURSEID`)**

**)**

**CREATE TABLE `grade\_report` (**

**`SID` int(10) NOT NULL,**

**`SECTIONID` int(11) NOT NULL,**

**`NUMBER\_GRADE` int(5) DEFAULT NULL,**

**`GRADE` varchar(2) DEFAULT NULL,**

**`GUIDE\_SSN` int(9) DEFAULT NULL,**

**PRIMARY KEY (`SID`,`SECTIONID`),**

**KEY `grade\_report\_ibfk\_2` (`GUIDE\_SSN`),**

**CONSTRAINT `grade\_report\_ibfk\_1` FOREIGN KEY (`GUIDE\_SSN`) REFERENCES `staff` (`SSN`),**

**CONSTRAINT `grade\_report\_ibfk\_2` FOREIGN KEY (`GUIDE\_SSN`) REFERENCES `staff` (`SSN`)**

**)**

**CREATE TABLE `prerequisite` (**

**`MAINCOURSEID` varchar(9) NOT NULL,**

**`PREREQCOURSEID` varchar(9) NOT NULL,**

**PRIMARY KEY (`MAINCOURSEID`,`PREREQCOURSEID`),**

**KEY `PREREQCOURSEID` (`PREREQCOURSEID`),**

**CONSTRAINT `prerequisite\_ibfk\_1` FOREIGN KEY (`MAINCOURSEID`) REFERENCES `course` (`COURSEID`),**

**CONSTRAINT `prerequisite\_ibfk\_2` FOREIGN KEY (`PREREQCOURSEID`) REFERENCES `course` (`COURSEID`)**

**)**

**CREATE TABLE `project` (**

**`PROJECTNAME` varchar(20) NOT NULL,**

**`SID` int(10) NOT NULL,**

**`PROJECTDESC` varchar(70) NOT NULL,**

**PRIMARY KEY (`PROJECTNAME`,`SID`),**

**KEY `SID` (`SID`),**

**CONSTRAINT `project\_ibfk\_1` FOREIGN KEY (`SID`) REFERENCES `student` (`SID`)**

**)**

**CREATE TABLE `thesis` (**

**`THESISID` int(11) NOT NULL,**

**`THESISNAME` varchar(50) NOT NULL,**

**`THESISDESC` varchar(70) NOT NULL,**

**`SSN` int(9) NOT NULL,**

**`SID` int(10) NOT NULL,**

**PRIMARY KEY (`THESISID`),**

**KEY `SSN` (`SSN`),**

**KEY `SID` (`SID`),**

**CONSTRAINT `thesis\_ibfk\_1` FOREIGN KEY (`SSN`) REFERENCES `staff` (`SSN`),**

**CONSTRAINT `thesis\_ibfk\_2` FOREIGN KEY (`SID`) REFERENCES `student` (`SID`)**

**)**

**CREATE TABLE `student\_ra` (**

**`SID` int(10) NOT NULL,**

**`OFFICELOCATION` varchar(20) DEFAULT NULL,**

**`GUIDE\_SSN` int(9) NOT NULL,**

**PRIMARY KEY (`SID`),**

**KEY `GUIDE\_SSN` (`GUIDE\_SSN`),**

**CONSTRAINT `student\_ra\_ibfk\_1` FOREIGN KEY (`GUIDE\_SSN`) REFERENCES `staff` (`SSN`)**

**)**

**CREATE TABLE `student\_ta` (**

**`SID` int(10) NOT NULL,**

**`SECTIONID` int(11) NOT NULL,**

**PRIMARY KEY (`SID`),**

**KEY `SECTIONID` (`SECTIONID`),**

**CONSTRAINT `student\_ta\_ibfk\_1` FOREIGN KEY (`SID`) REFERENCES `student` (`SID`),**

**CONSTRAINT `student\_ta\_ibfk\_2` FOREIGN KEY (`SECTIONID`) REFERENCES `section` (`SECTIONID`)**

**)**

**Most frequently used SQL queries:**

**1) Authentication query**

(Checking whether an staff has submitted the correct NetID and Password combination and then allowing entry into the system)

SELECT PASSWORD FROM STAFF WHERE NETID = ‘rbk122130’;

(Checking whether an student has submitted the correct NetID and Password combination and then allowing entry into the system)

SELECT PASSWORD FROM STUDENT WHERE NETID = ‘rxl122130’;

**2) Queries to insert a course:**

INSERT INTO GRADE\_REPORT VALUES (2021147361,83333,NULL,NULL,871632456)

**3) Query to drop a course:**

DELETE FROM GRADE\_REPORT WHERE SID = ‘2021147361’ AND SECTIONID =’ 83333’;

**4) Query to view the students**

SELECT G.SID AS SID,s.fname as FNAME,s.lname as LNAME,g.grade as GRADE

FROM grade\_report G, STUDENT S, SECTION E

WHERE G.SID =S.SID AND G.SECTIONID = E.SECTIONID

AND E.COURSEID = ‘6350’

AND E.SECTIONNO = ‘501’;

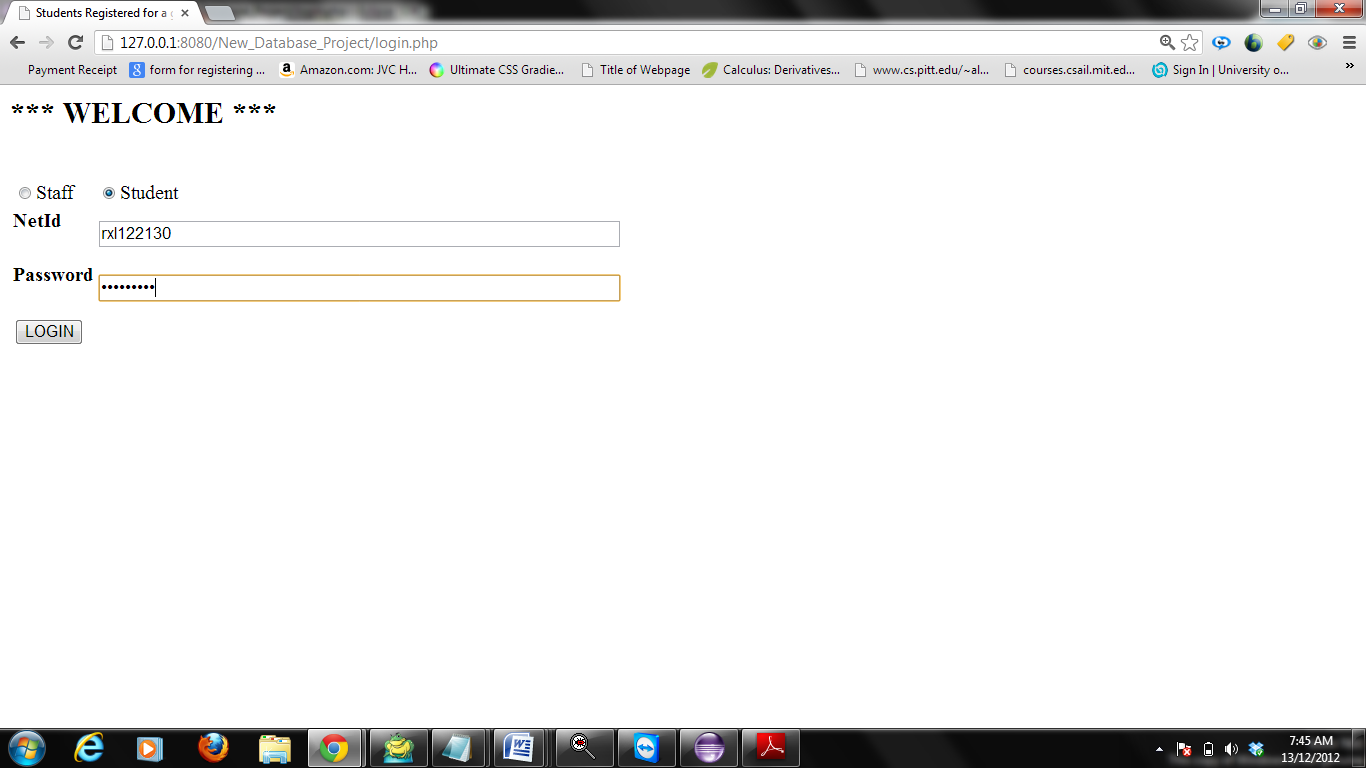
**5)Query to view available courses**

SELECT S.STATUS AS CLASS\_STATUS,S.SEMESTER AS SEMESTER,S.YEAR AS YEAR,S.SECTIONID AS CLASS\_NO,S.COURSEID AS COURSE\_NO,S.SECTIONNO AS SECTION\_NO,C.COURSENAME AS COURSE\_NAME,S.LECTURE\_DAY AS LECTURE\_DAY,S.START\_TIME AS CLASS\_START\_TIME,S.END\_TIME AS CLASS\_END\_TIME,S.LOCATION AS CLASS\_LOCATION,F.FNAME AS INSTRUCTOR\_FNAME,F.LNAME AS INSTRUCTOR\_LNAME,C.CDESCRIPTION AS CDESCRIPTION,S.CREDIT\_HOURS AS CLASS\_CREDIT\_HOURS FROM SECTION S, COURSE C, STAFF F WHERE F.SSN = S.INSTRUCTORSSN AND S.COURSEID = C.COURSEID AND S.SEMESTER LIKE '%Fall%' AND C.COURSENAME LIKE '%database%' AND C.COURSEID LIKE '%81111%' AND F.SSN IN (SELECT SSN FROM STAFF WHERE FNAME LIKE '%balaji%' OR LNAME LIKE '%balaji%') AND LECTURE\_DAY like '%tue%' AND LECTURE\_DAY like '%thur%'

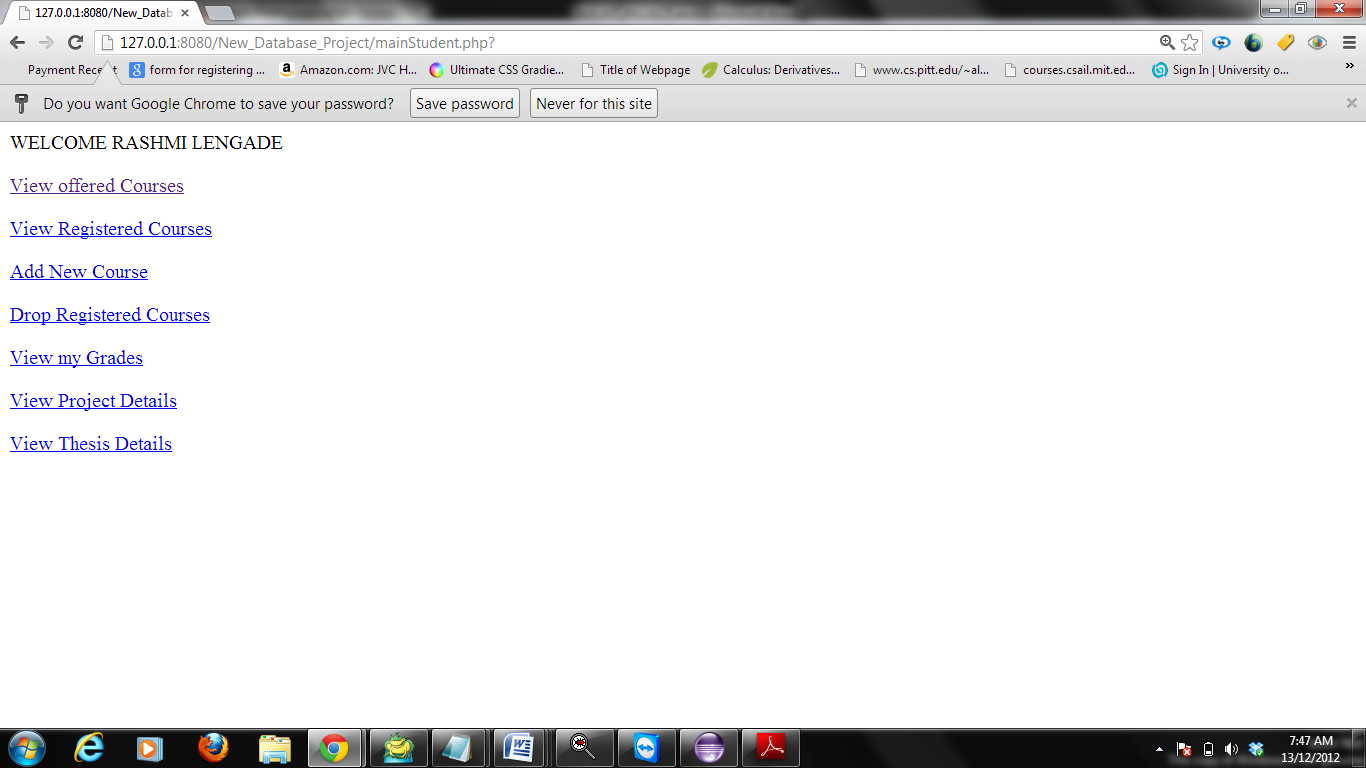
INSERT INTO Feedback VALUES ($CRN,$CLO\_No,$below,$progress,$meet,$exceed,'$material')

**Sample Screen Shots:**

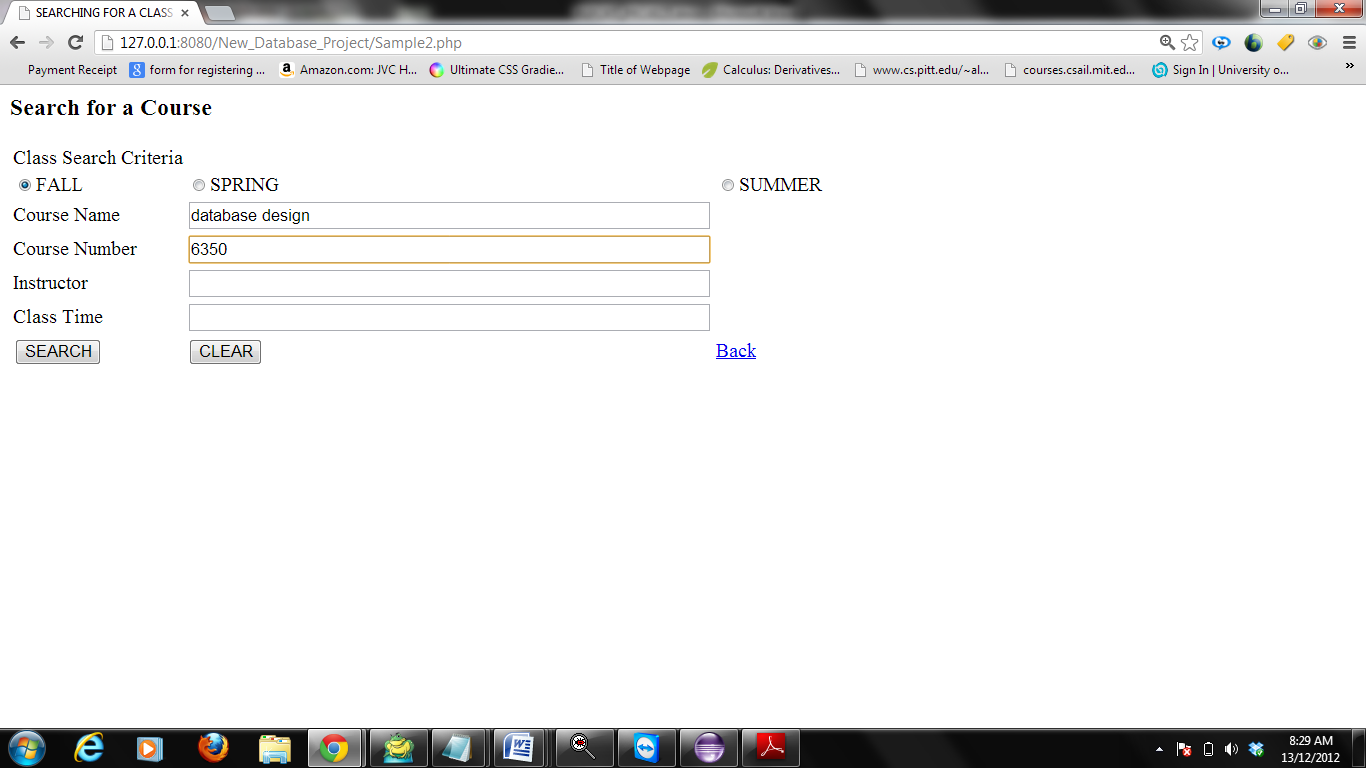
**1) Login Screen**

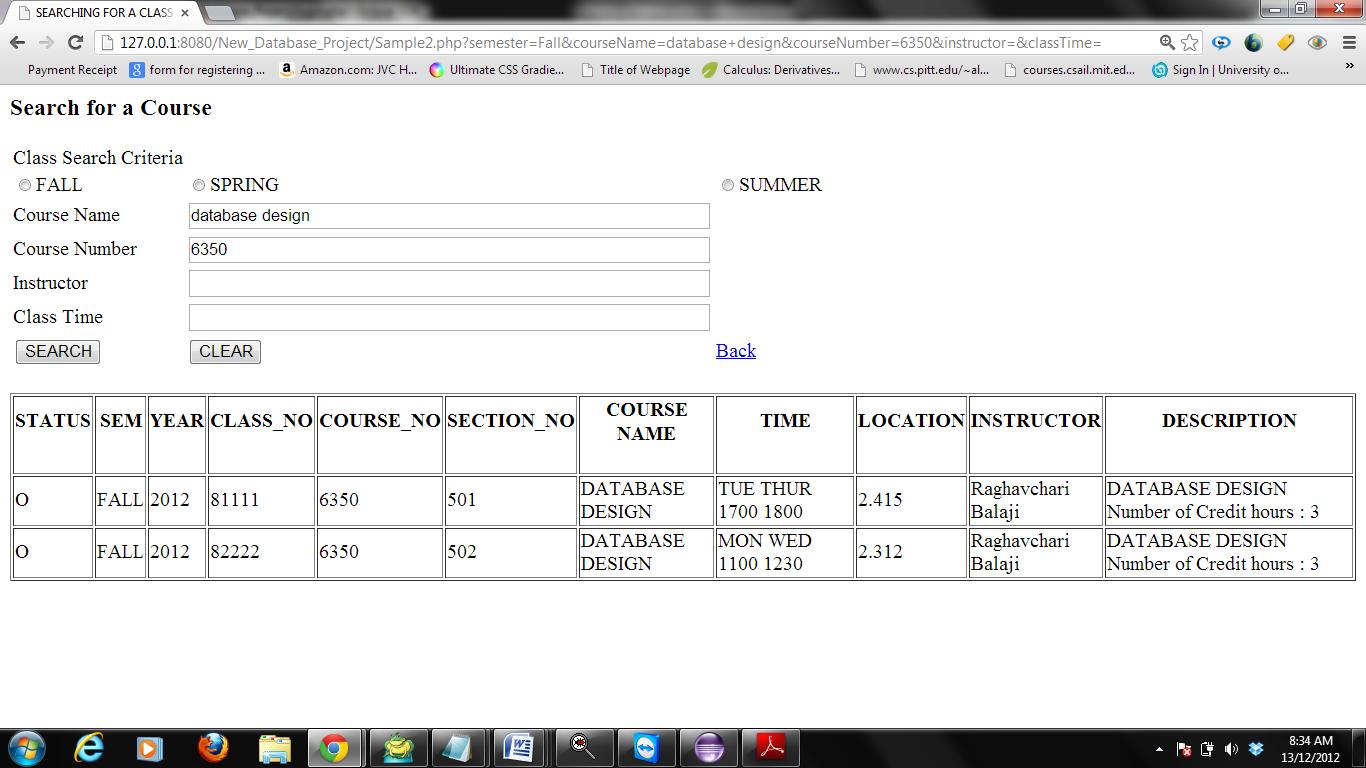
****

**2) Student Login successfully and reached the main page**

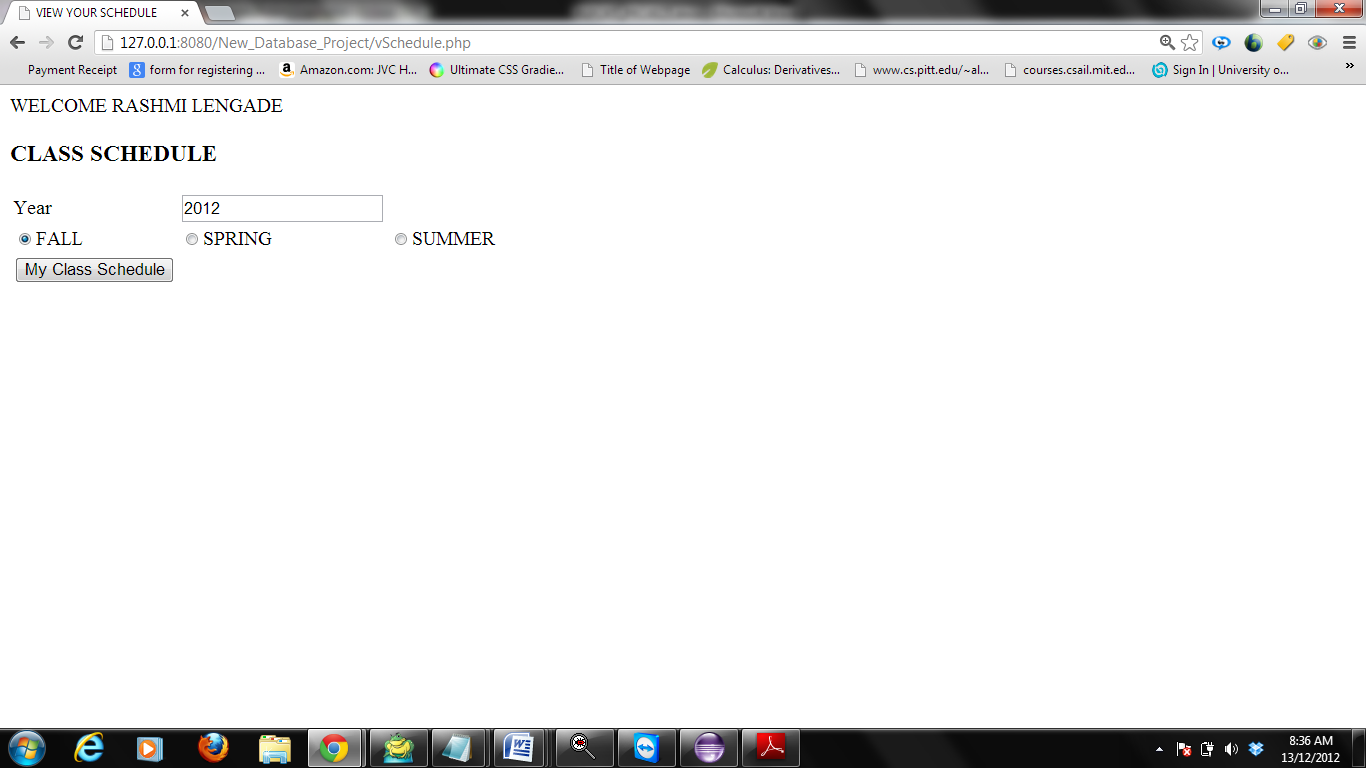
****

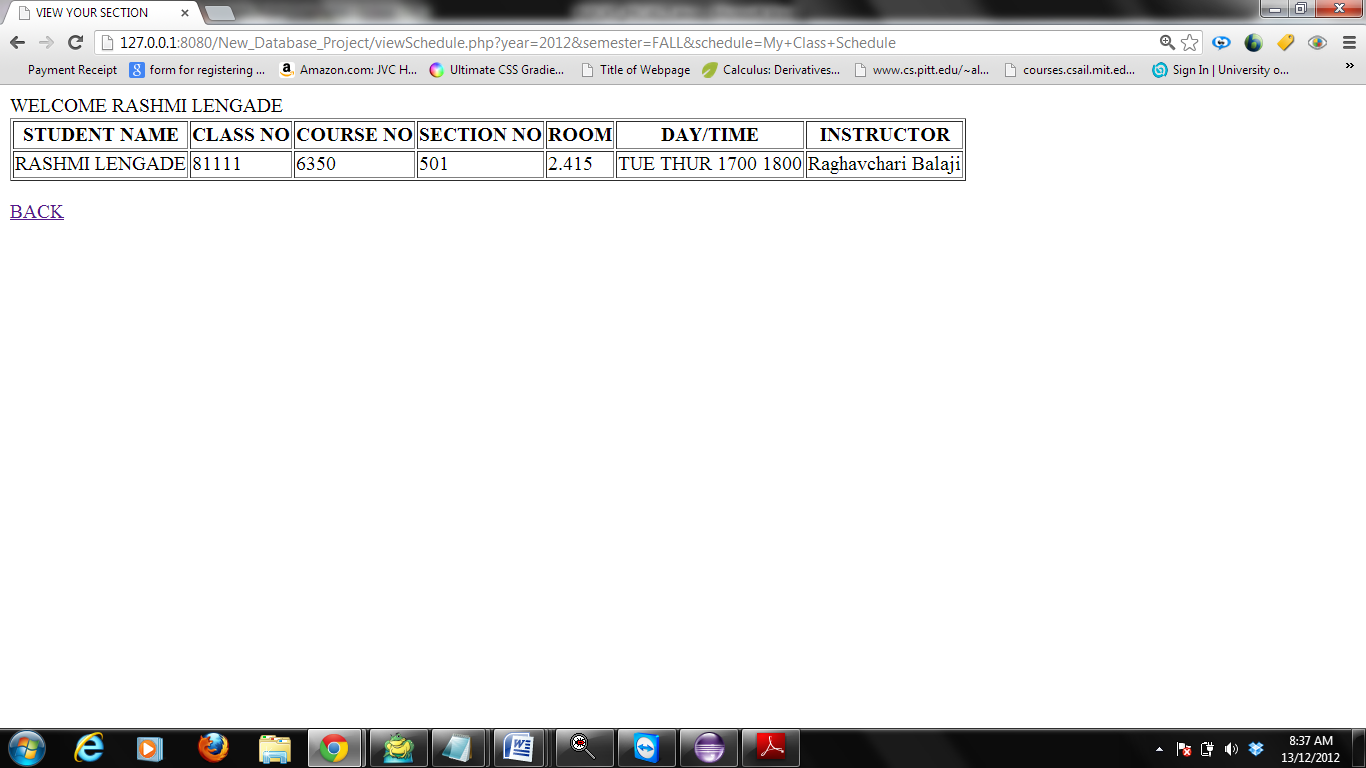
**3) Student can view the offered courses by looking into the Course lookup:**

****

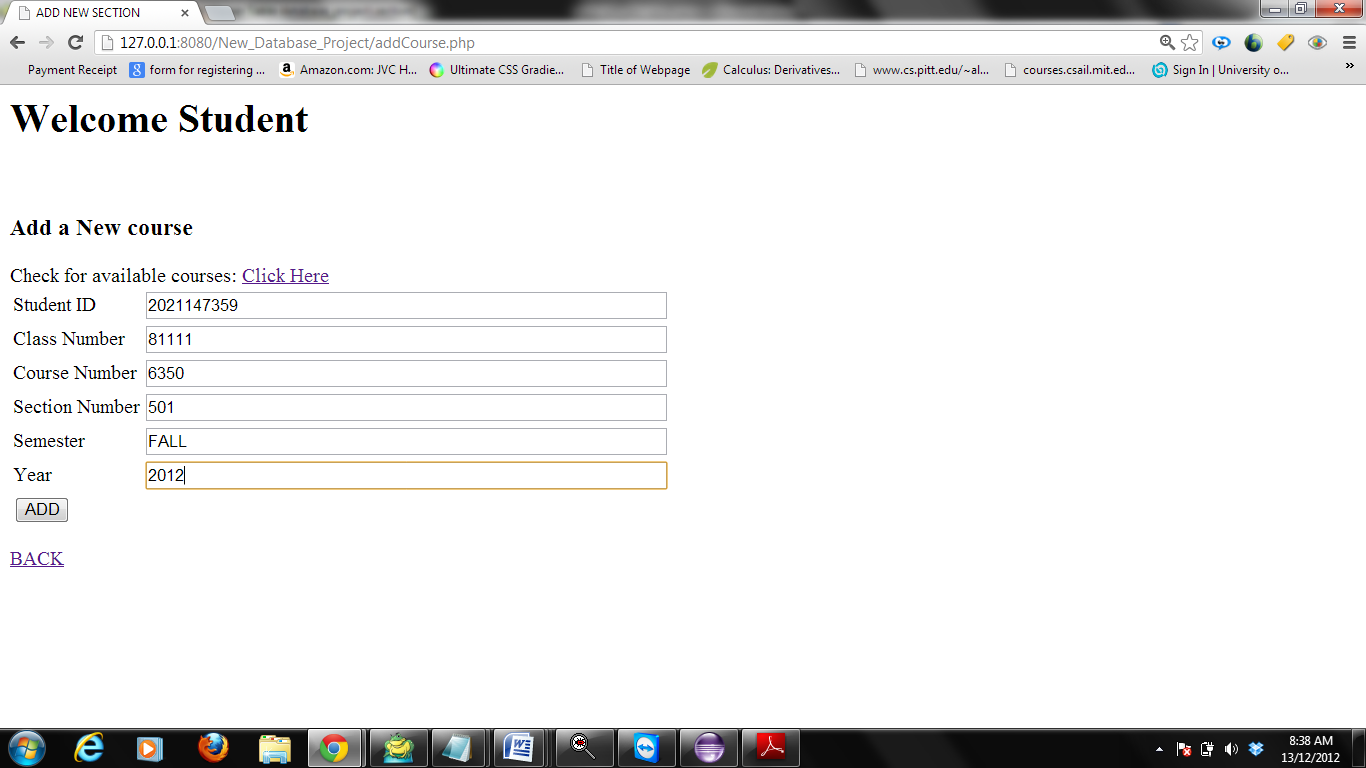
****

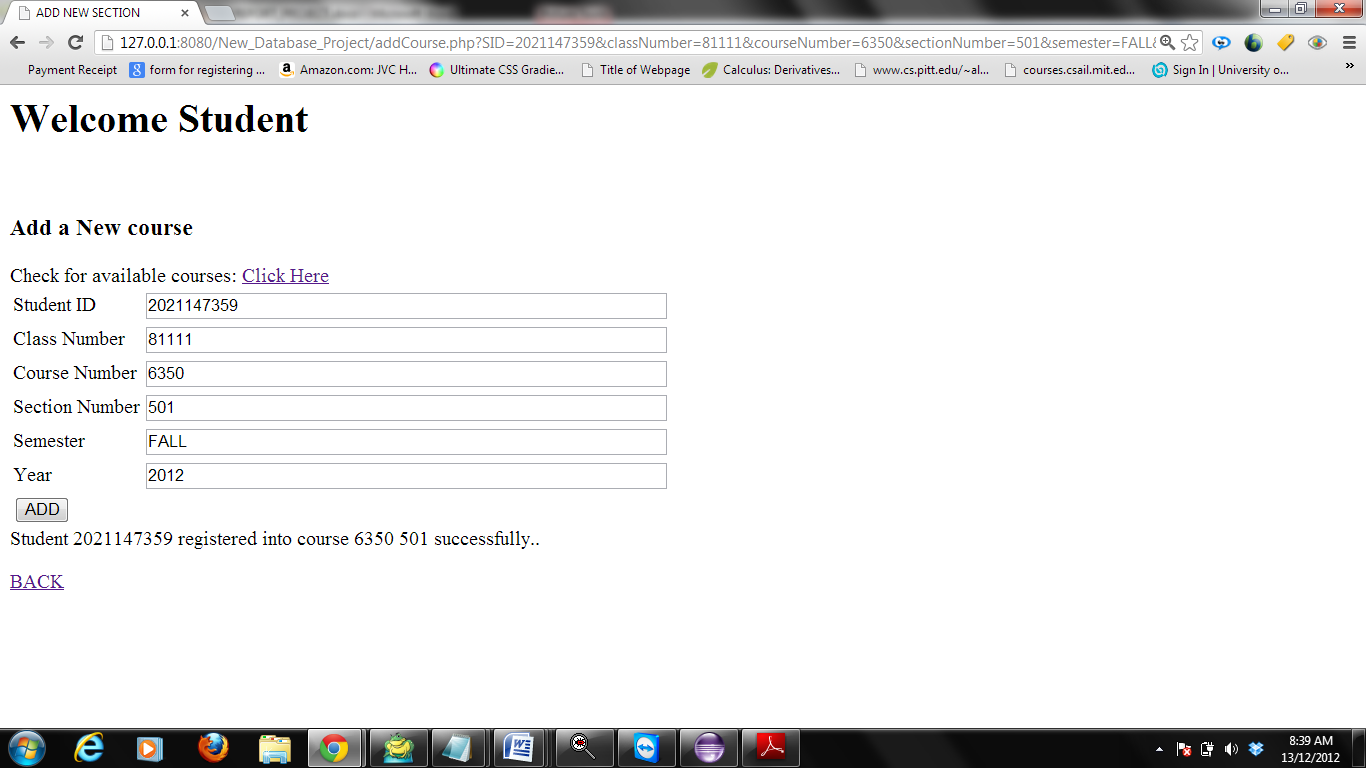
**4) Student can view the courses into which he/she is registered**



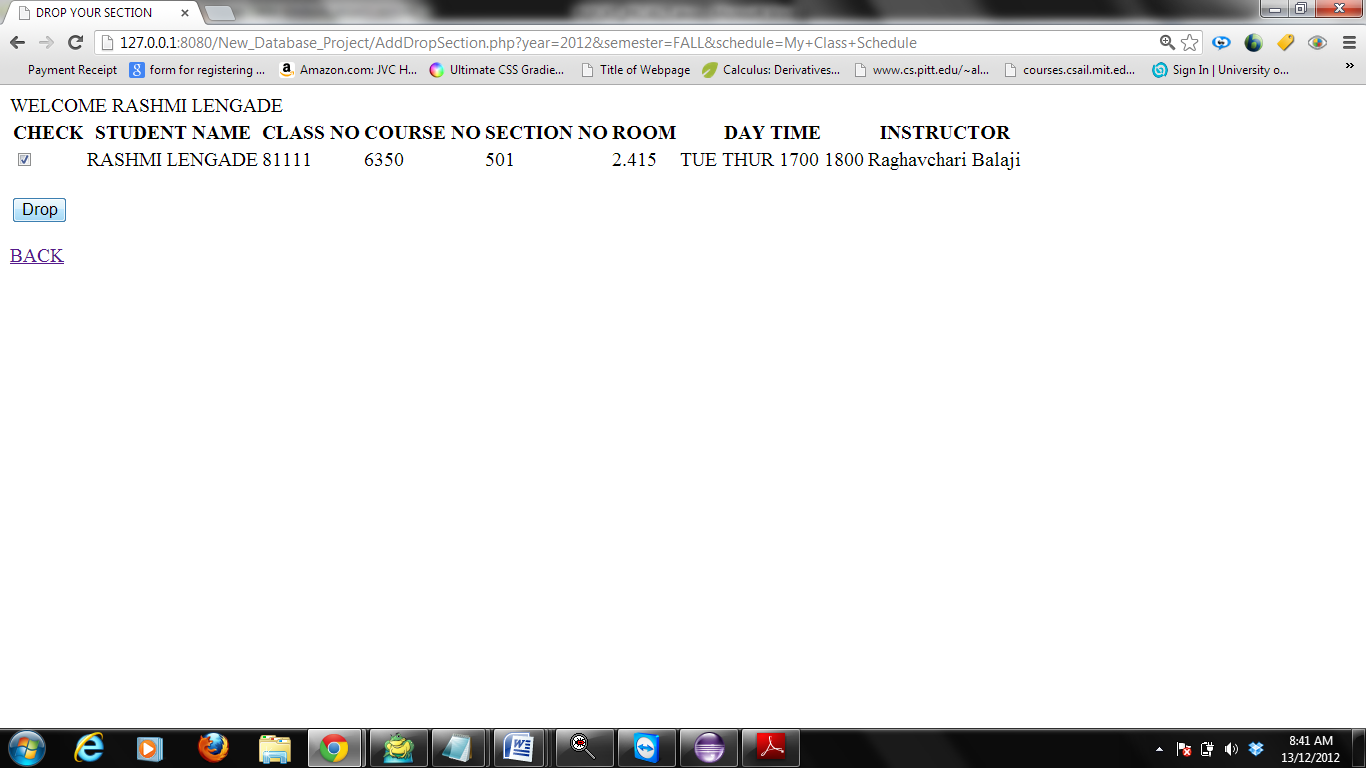
****

**4) Student can add a course to his schedule**

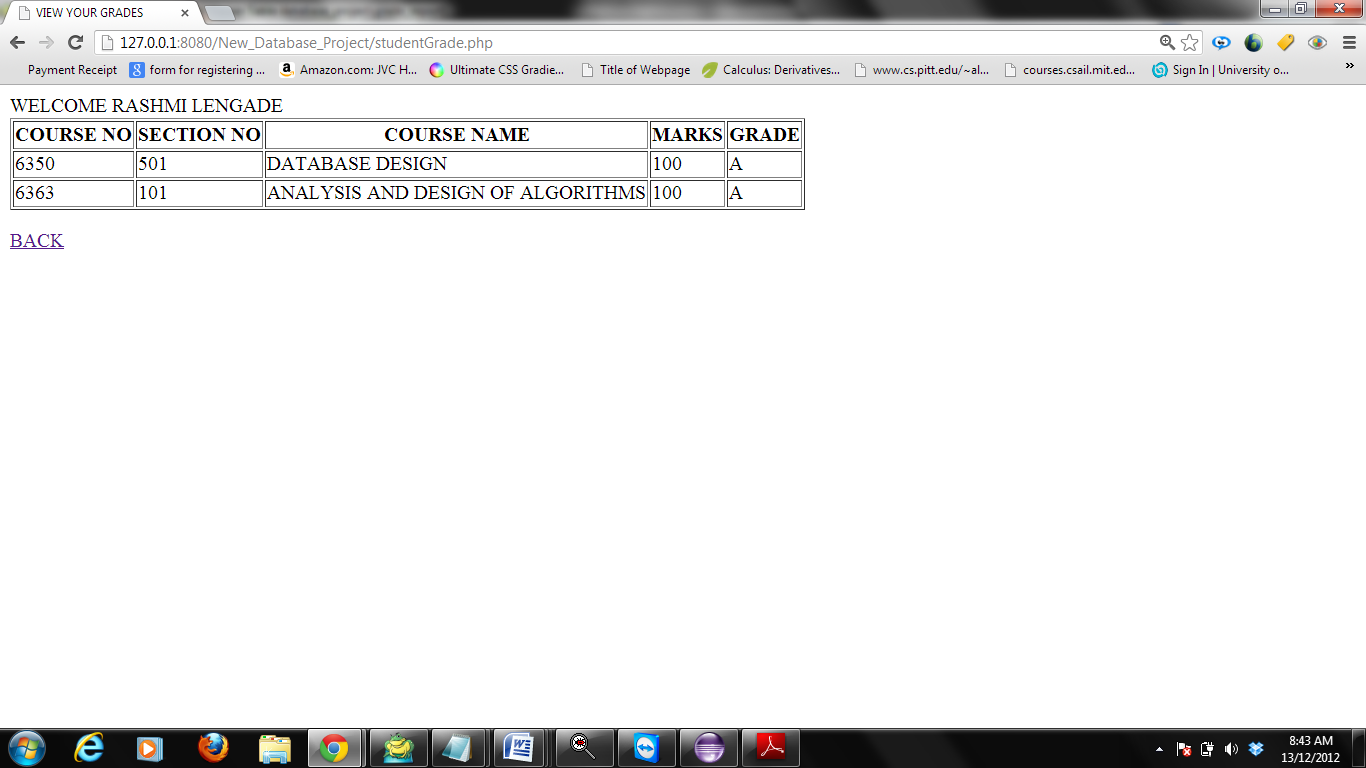




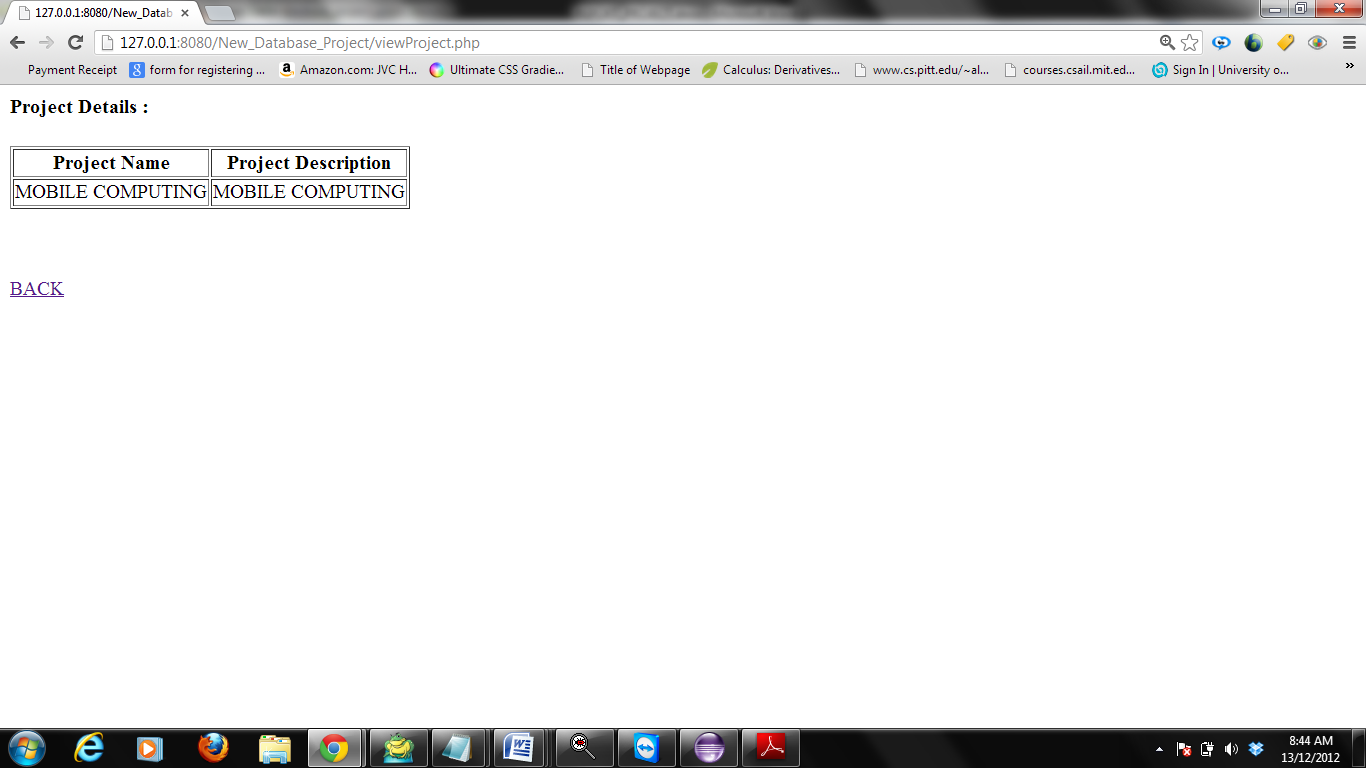
**5) Student can Drop a course into which he /she has already registered**

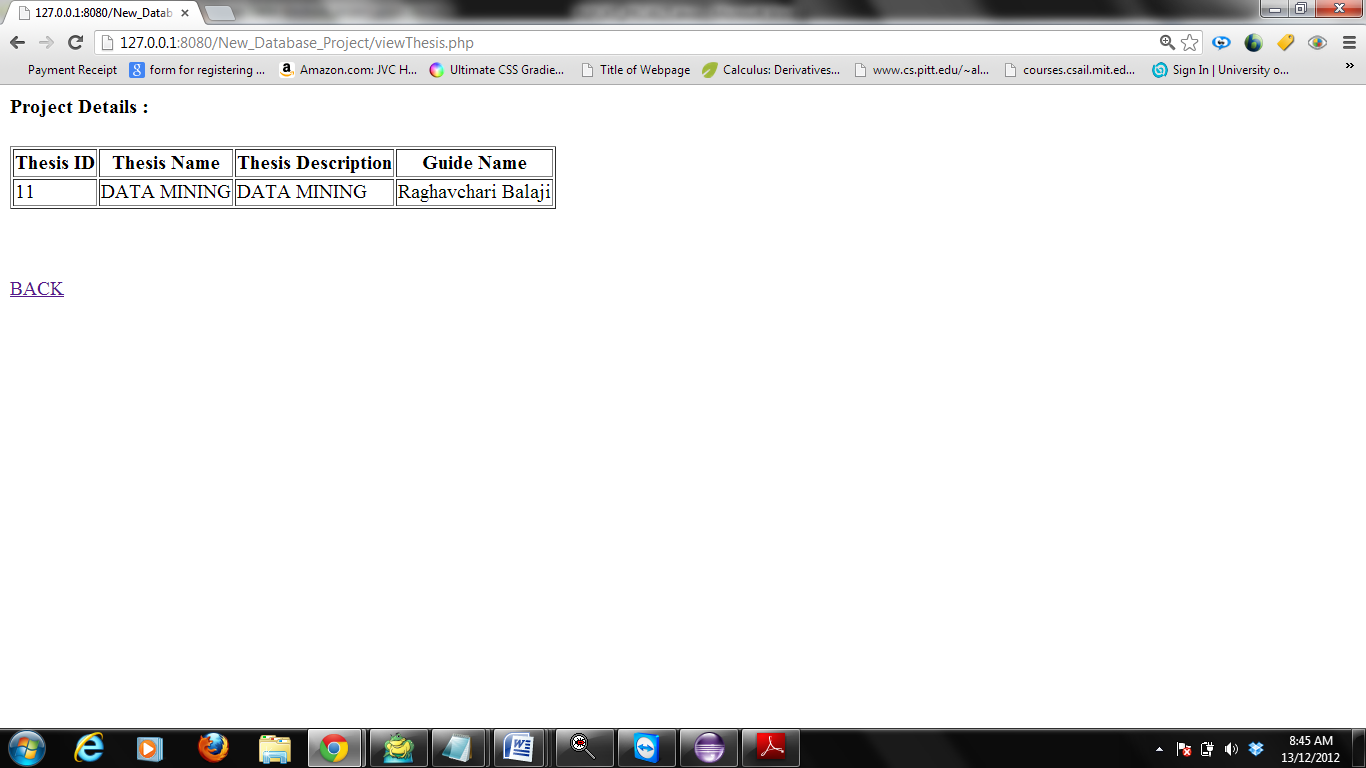


**6) Student can view the grade of the courses into which he is registered:**

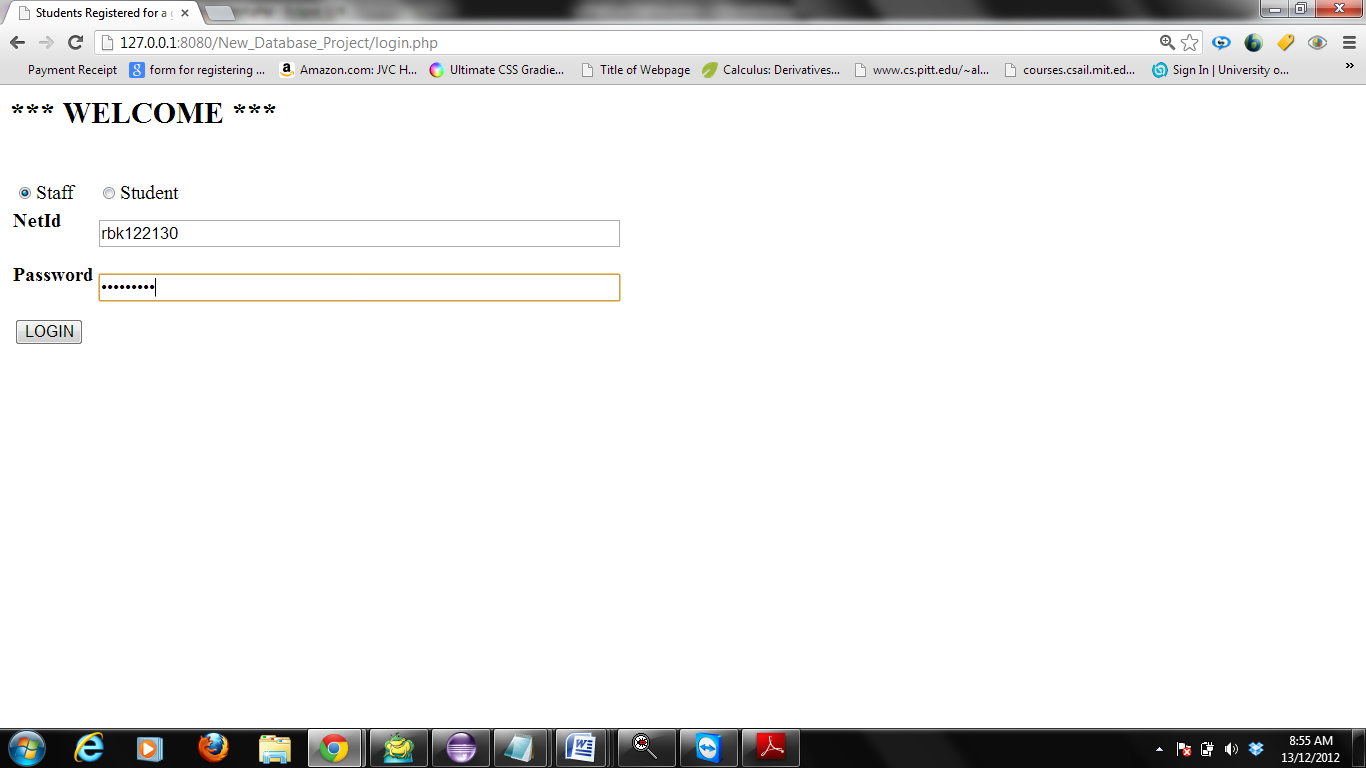
****

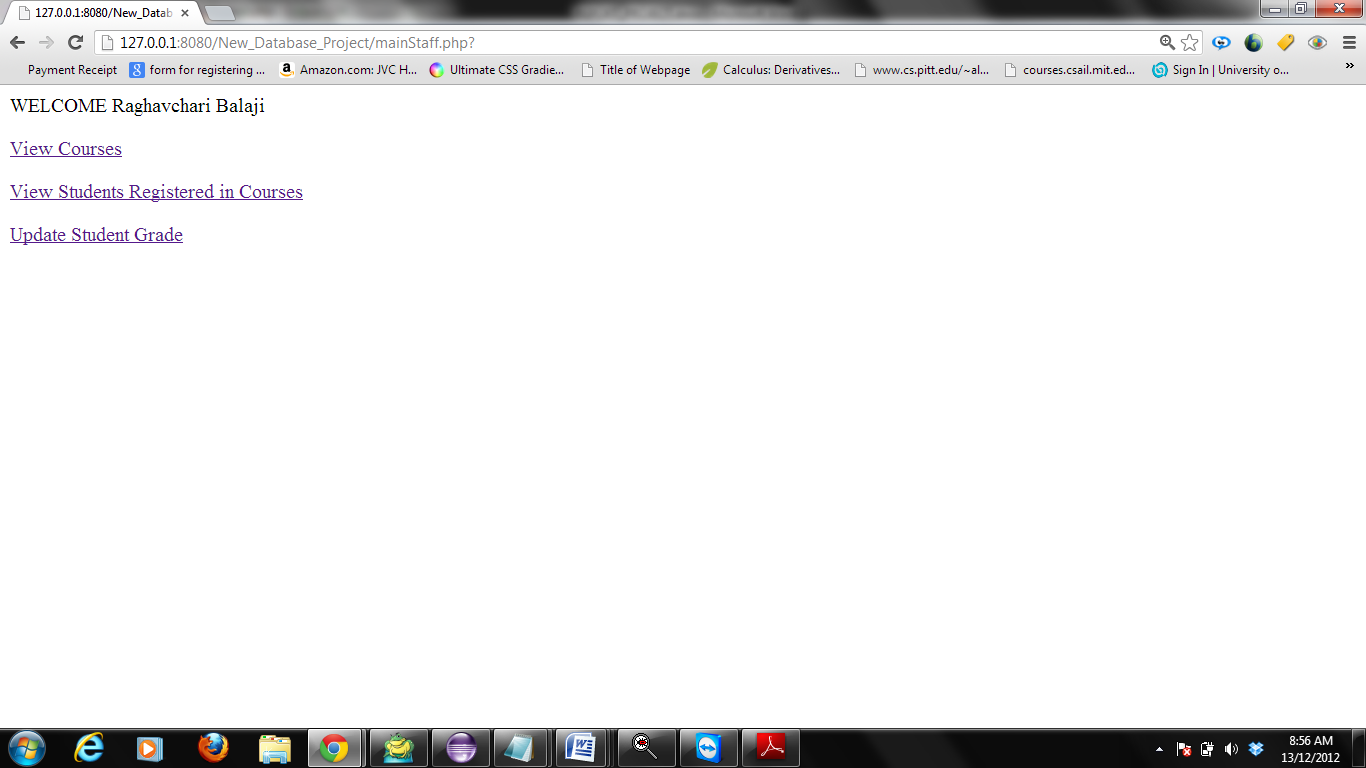
**7)Student can view the details of the thesis and project assignment to him**

****

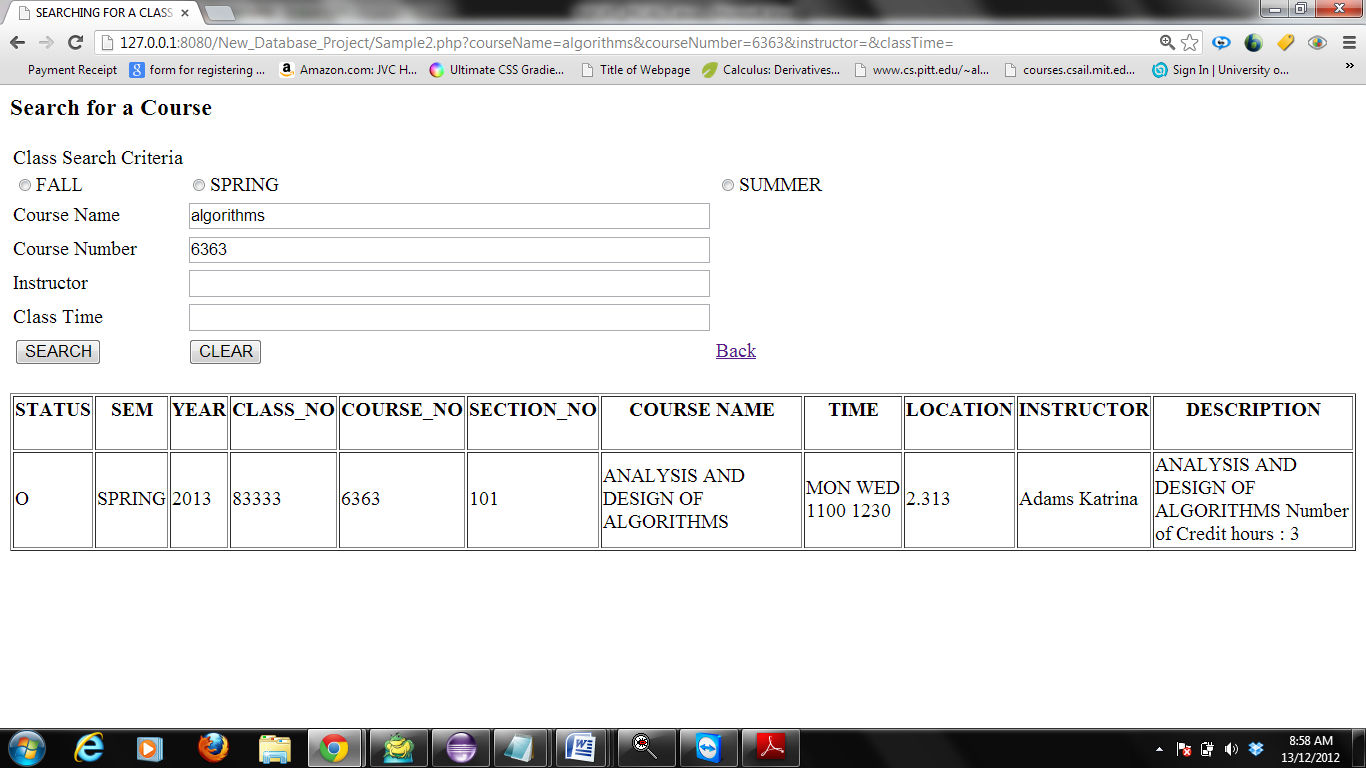
****

**8) Staff login**

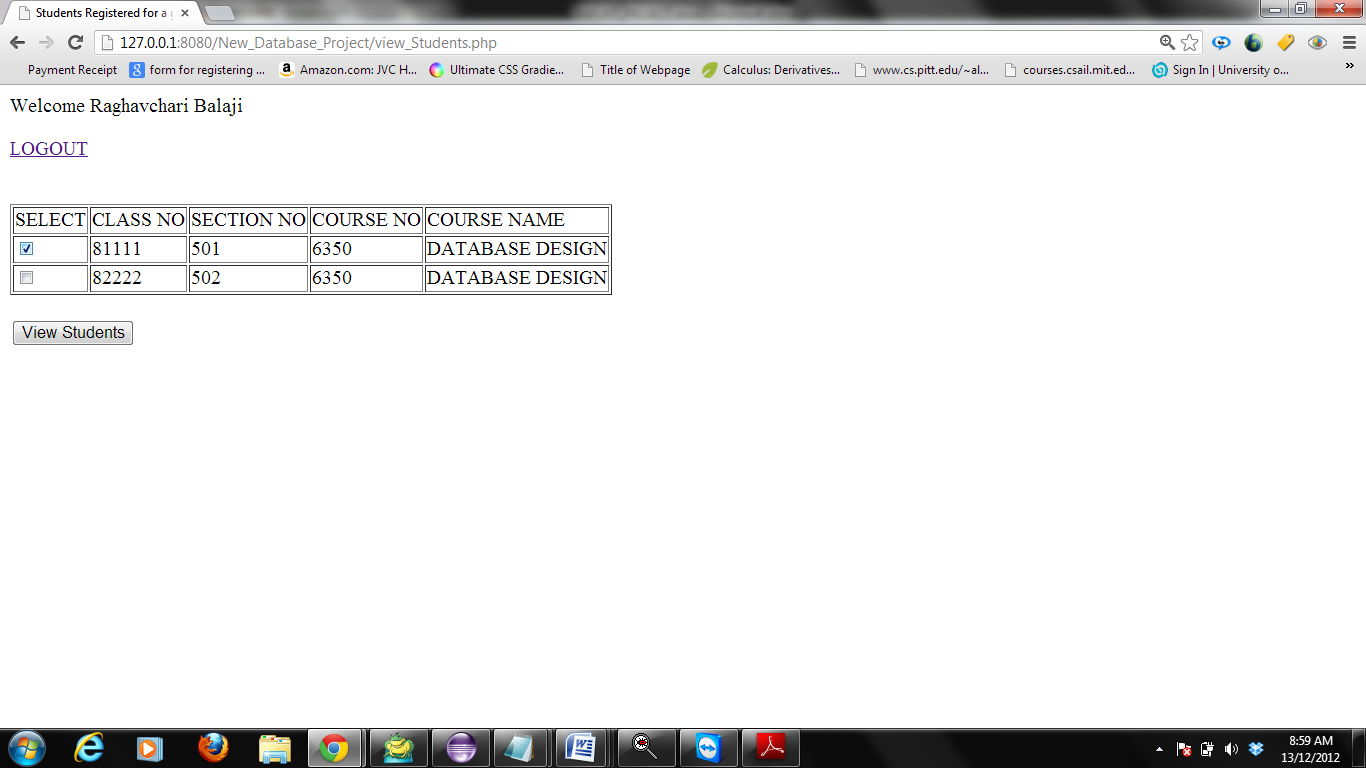
****

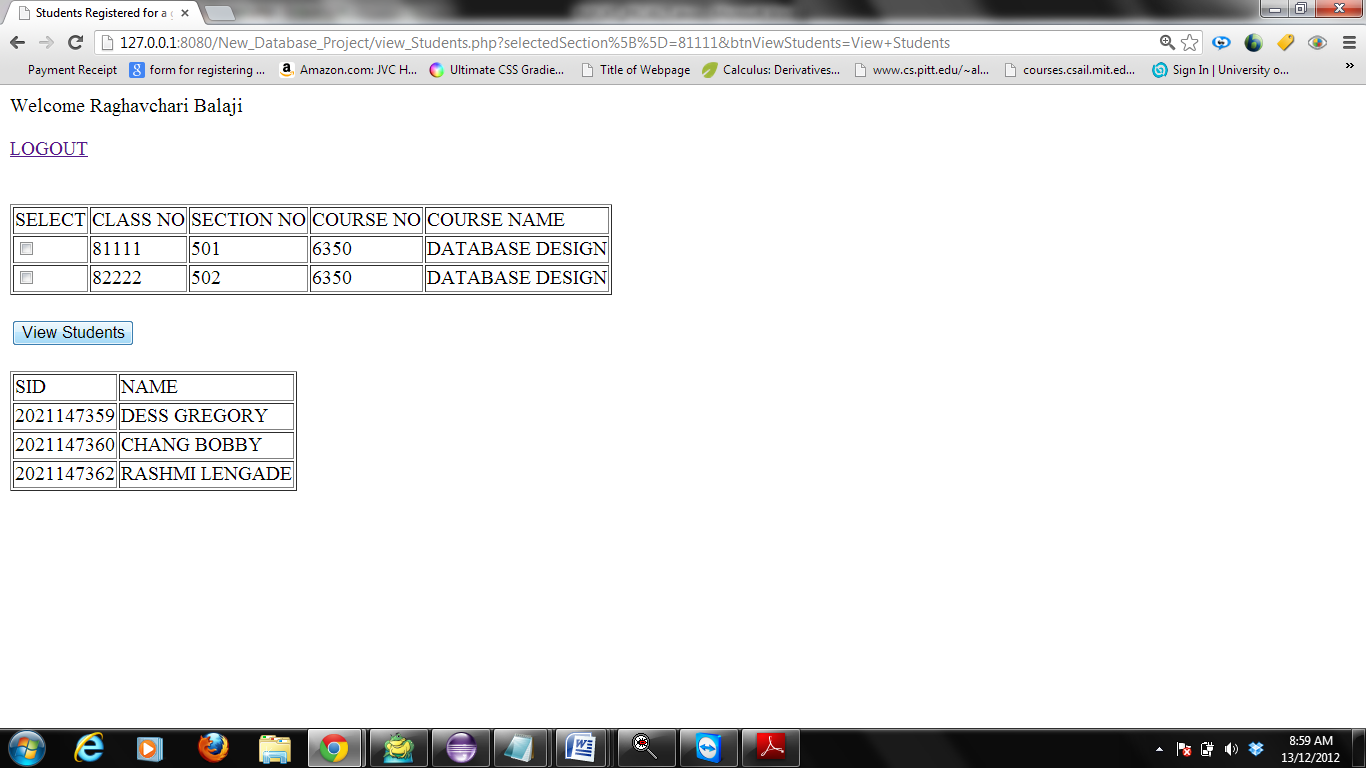
****

**9) Staff can view the course lookup**

****

**10) Staff can view the students registered under him/her**

****

****

**Extra Credit-Work:**

1. Student can login to the system using their NETID and Password.
2. Staff can login to the system using their NETID and Password.
3. Before dropping a section student is able to view the entire registered course list.
4. Student can view all the grades for all the registered courses.
5. Student can view the details of the thesis he/she is assigned.
6. Student can view the details of the project he/she is assigned.
7. Staff can go through the course lookup for looking into the courses offered and the course timings to get an idea to set his class timing.

**Conclusion:**

The deliverable includes

The Web GUI for the end users to interact with the system.

The Login capability based on type of user (Student/staff) for secured access to the system.

The functional requirements which include course lookup based on search fields like semester, course number, course name, staff and class time.

A Student can register for a course only once by adding the required details like the section and course number, student ID.

A Student can drop a course.

Viewing list of students who have registered for a particular section (Staff’s Home page)

**Future Scope:**

Instructor should be able to display information about the course to the students.

Instructor should be able to send emails to students about the updates.

Student should be assigned academic advisor.

Student should be able to view class details like assignments and exam schedule.

Admin should be created who can add data to the system and also should be able to add an instructor.

**References:**

1. <http://www.w3schools.com/>

2. <http://us3.php.net/manual/en/book.session.php4.http://www.mysql.com/>

3. <http://dev.mysql.com/doc/refman/5.0/en/example-auto-increment.html>

**Collaborators :** Yogesh Jagtap

Rahul Gaikwad