**CS 6360 DATABASE DESIGN**

**PROJECT REPORT**

**Submitted to**

**Prof. Balaji Raghavachari**

**By**

**Rekha Muthulakshmi Nachadalingam**

[rxn121330@utdallas.edu](mailto:rxn121330@utdallas.edu)

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **SNo** | **Content** | **Page No** |
| **1.** | **Introduction** | **1** |
| **2.** | **Data and Functional Requirements** |  |
|  | **2.1 Data Requirements** | **1** |
|  | **2.2 Entities and Attributes** | **2** |
|  | **2.3 Relationships between Entities** | **2** |
|  | **2.4 Functional Requirements** | **4** |
| **3.** | **Design of the Database Schema** |  |
|  | **3.1 EER Diagram of the University Database** | **7** |
|  | **3.2 Mapping to Relational Model** | **8** |
| **4.** | **Functional Dependencies and Normalization** | **9** |
| **5.** | **SQL Queries** |  |
|  | **5.1 Creation of Tables and Primary Key Constraints** | **12** |
|  | **5.2 Creation of Foreign Key Constraints** | **15** |
| **6.** | **Data in the tables** | **16** |
| **7.** | **Stored Procedures created to Implement the System** | **21** |
| **8.** | **Demonstration of the Application** | **26** |
| **9.** | **Additional Functionality Implemented** | **32** |
| **10.** | **Conclusion** | **35** |
| **11.** | **References** | **35** |

**1. INTRODUCTION**

The application developed is a web application for a University and is used to maintain information about students, courses and faculty. The target users are the students and faculty of the University and access is restricted only to them. It provides various features such as Course Lookup, Course Registration, Viewing the list of students registered in a particular section, Viewing and editing a student’s contact details, Viewing student grades, Viewing the class schedule of a student, faculty details, etc. All of these features are available to both the students and instructors of the University. The following are the technologies used to develop the application:

* **Front End:** ASP.NET C# in Visual Studio 2012
* **Back End:** Microsoft SQL Server 2008 R2

**2. DATA AND FUNCTIONAL REQUIREMENTS**

**2.1 DATA REQUIREMENTS**

The University database keeps track of 2 types of people: Students and Employees. A person may belong to one or both the types. Each person is uniquely identified by his/her NetID. The Name, Address, Sex, SSN and Phone Number of a person are stored. Although SSN is also unique for each person, it is possible that some students may not have a SSN. Hence it is not a key attribute.

1. Each student has a Student Number, Degree and a Major. A student may also be a Student Worker.
2. Every employee has a salary and Employee type. Employees may belong to one of the following 2 types: Instructors and Student Workers. Instructors have a rank, office, office hours and may belong to one or more departments. Student Workers have a particular number of Working hours and are further classified into 2 types: Research Assistant and Teaching Assistant. Research Assistant has the research project he/she is working on. Teaching Assistant has the course he/she works on.
3. Each department is identified by a Department Number and has a department name and the college that has the department. A department may offer one or more courses.
4. Each course has a Course Number, Course Name, Description, Level, Credit hours and the department that offers the course. Course Number uniquely identifies each course.
5. Each section has a section number, course, semester, year and an instructor. A section is uniquely identified by the combination of course, section number, year and semester. A unique ClassNumber is assigned to each such combination for the purpose of simplification. Hence Class Number is the key to uniquely identify a section.
6. A section is taught by an instructor in a particular classroom in a particular time. Each section has a limit on the maximum number of seats available for that semester. It is denoted by Totalseats. The number of seats currently available is denoted by AvailSeats.
7. The student’s grade is maintained for each course he/she is enrolled.

**2.2 ENTITIES AND ATTRIBUTES:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ENTITY** | **ATTRIBUTES** | **PRIMARY KEY** | **FOREIGN KEY** |
| Person | NetID, Name, Address, Sex, SSN and Phone Number | NetID |  |
| Student | NetID, StudentNo ,Degree | NetID | NetID (References Person) |
| Employee | NetID, emp\_type, salary | NetID | NetID (References Person) |
| Instructor | NetID, rank, office, office hours | NetID | NetID (References Employee) |
| StudentWorker | NetID ,Hours, type | NetID | NetID (References Employee) |
| Research Assistant | NetID ,Research topic | NetID | NetID (References StudentWorker) |
| Teaching Assistant | NetID ,Course | NetID | NetID (References StudentWorker) |
| Department | DNo, DName, College | DNo |  |
| Course | CourseNo, Course Name, Description, Level, Credits | CourseNo |  |
| Section | Class Number, Section number, CourseNo, semester, year | Class Number | CourseNo (References Course) |

**2.3 RELATIONSHIPS BETWEEN ENTITIES:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ENTITY 1** | **RELATIONSHIP** | **ENTITY 2** | **ATTRIBUTES OF THE RELATIONSHIP** |
| Student | Majors in | Department |  |
| Instructor | Belongs to | Department |  |
| Instructor | Teaches | Section | Lecture Time, Class Room, Total seats, Available seats. |
| Course | Offered by | Department |  |
| Course | Has | Section |  |
| Student | Enrolls in | Section | Grade |

**2.4 FUNCTIONAL REQUIREMENTS**

**2.4.1.Course Lookup**

User must be able to search for the sections being taught in a given semester through search parameters like course name, semester, instructor, course number, lecture time, semester, year. The search must display all the course sections that match the search criteria.

**Main Flow:**

1. User navigates to the Course Lookup page.
2. User specifies at least one of the following search parameters and submits.
   1. Year
   2. Semester
   3. Course Name
   4. Course Number
   5. Instructor
   6. Lecture Start Time
   7. Lecture End Time
3. System displays the details of the sections that satisfy the search conditions.

**Alternate Flow:**

2a. User does not specify any of the search parameters.

1. System prompts the user to specify at least one parameter.
2. Return to Main Flow Step 2.

**2.4.2.Course Registration**

User must be able to enroll or drop a student from a course section. To enroll a student to a section, the availability of seats must be checked and if the student is not already enrolled in that class, he/she must be added to the class and the course must be added to the student’s course list. Similarly, when a student drops from a class, the course must be removed from the student’s course list.

**Main Flow:**

1. User navigates to the Course Registration Page.
2. User enters the Class Number to which a student must be enrolled or dropped.
3. System validates the Class Number.
4. System displays the class details such as Instructor’s ID, lecture days, lecture time and the number of available seats.
5. User enters the Net ID of the student and selects whether to Enroll or Drop.
6. System validates the Net ID of the student.
7. System validates that the student is not already enrolled to this class (for enrollment) / the student is currently enrolled in this class (for dropping).
8. System checks the availability of seats.(for enrollment)
9. System enrolls/drops the student to/from the class and informs the user.

**Alternate Flow:**

3a. Class Number is invalid.

1. System informs the user and prompts to enter a valid Class Number.
2. Return to Main Flow Step 2.

6a. Student Net ID is invalid.

1. System informs the user and prompts to enter a valid Net ID.
2. Return to Main Flow Step 5.

7a. Student is already enrolled in the specified class.(for enrollment)

1. System informs the user that the student is already enrolled in the class.
2. Return to Main Flow Step 5.

7b. Student is not currently enrolled in this class. (for dropping)

1. System informs the user that the student is not enrolled in the class.
2. Return to Main Flow Step 5.

8a. There are no seats available in the class.

1. System informs the user that no seats are available.
2. Return to Main Flow step 2.

**2.4.3.View list of students in a section**

User must be able to view the names of the students who are enrolled in a particular section by specifying the Class Number of the section.

**Main Flow:**

1. User Navigates to the View Students page.

2. User enters the Class Number of the section.

3. System validates the Class Number.

4. System displays the list of Students who have been enrolled to the specified section.

**Alternate Flow:**

3a. Class Number is invalid.

1. System informs the user and prompts to enter a valid Class Number.
2. Return to Main Flow Step 2.

**2.4.4.View Student profile**

User must be able to view a student’s profile by specifying the student’s NetID .The search must display the student’s details including the courses enrolled and grades.

**Main Flow:**

1. User enters the Net ID of the student and submits.
2. System validates the Net ID.
3. System displays the courses that the student is enrolled in along with the grade.

**Alternate Flow:**

2a. Student Net ID is invalid.

1. System informs the user and prompts to enter a valid Net ID.
2. Return to Main Flow Step 1.

**2.4.5.View class schedule**

User must be able to view the class schedule of a particular student. For each course that the student is registered, the system must display the class room, lecture days, lecture timing.

**Main Flow:**

1. User enters the Net ID of the student and submits.
2. System validates the Net ID.
3. System displays the class schedule of the student.

**Alternate Flow:**

2a. Student Net ID is invalid.

1. System informs the user and prompts to enter a valid Net ID.
2. Return to Main Flow Step 1.

**2.4.6.View Faculty Information**

User must be able to view the details of an instructor by specifying the instructor’s last name. The system must display the instructor’s Net ID, rank, office hours, office room.

**Main Flow:**

1. User enters the last name of the Instructor and submits.
2. System validates the last name.
3. System displays the details of the Instructor.

**Alternate Flow:**

1. System informs the user and prompts to enter a valid name.
2. Return to Main Flow Step 1.

**2.4.7.View/Update Student’s personal information**

User must be able to view as well as change the personal information of a Student such as Name, address, phone number, etc.

**Main Flow:**

1. User enters the Net ID of the student and submits.
2. System validates the Net ID.
3. System displays the details of the student along with the option to edit.
4. User edits the Student’s details and submits.
5. System updates the information and informs the user.

**Alternate Flow:**

2a. Student Net ID is invalid.

1. System informs the user and prompts to enter a valid Net ID.
2. Return to Main Flow Step 1.

**3.2 MAPPING TO RELATIONAL MODEL:**

1. **PERSON**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **NetID** | **SSN** | **FName** | **LName** | **Address** | **Sex** | **Phone** |

1. **STUDENT**

|  |  |  |  |
| --- | --- | --- | --- |
| **NetID** | **SNo** | **Degree** | **MajorDNo** |

1. **EMPLOYEE**

|  |  |  |
| --- | --- | --- |
| **NetID** | **Salary** | **Emp\_type** |

1. **INSTRUCTOR**

|  |  |  |  |
| --- | --- | --- | --- |
| **NetID** | **Rank** | **Office** | **Officehrs** |

1. **DEPARTMENT**

|  |  |  |
| --- | --- | --- |
| **DNo** | **DName** | **College** |

1. **STUDENTWORKER**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **NetID** | **Hours** | **RAflag** | **ResearchTopic** | **TAflag** | **CourseNo** |

1. **FACULTYDEPT**

|  |  |
| --- | --- |
| **NetID** | **DNo** |

1. **COURSE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CNo** | **CName** | **Description** | **Credits** | **Level** | **OfferingDNo** |

1. **SECTION**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ClassNumber** | **CNo** | **SectionNo** | **Semester** | **Year** | **FacultyNetID** | **LecDays** | **ClassRoom** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **InstructionMode** | **TotalSeats** | **AvailSeats** | **LecStartTime** | **LecEndTime** |

1. **STUDENTCOURSE**

|  |  |  |
| --- | --- | --- |
| **NetID** | **ClassNumber** | **Grade** |

**4. FUNCTIONAL DEPENDENCIES AND NORMALIZATION:**

**1.Person( NetID, SSN, FName, LName, Address, Sex, Phone)**

NetID → SSN FName LName Address Sex Phone

SSN → NetID FName LName Address Sex Phone

SSN, FName, LName, Address, Sex, Phone appear in RHS but never in LHS. Hence, they do not appear in any key.

NetID⁺ = { NetID, SSN, FName, LName, Address, Sex, Phone}

SSN⁺ = { NetID, SSN, FName, LName, Address, Sex, Phone }

So, the candidate keys are NetID and SSN.

**Prime:** {NetID, SSN} **Non Prime :** { FName, LName, Address, Sex, Phone }

There are no 3NF violations since no non-prime attribute is transitively dependent on the key.

Hence the table is in 3 NF.

**2.Student (NetID, SNo,Degree, MajorDno)**

NetID → SNo Degree MajorDno

SNo → NetID Degree MajorDno

Degree and MajorDno appear in RHS but not in LHS. Hence, they do not appear in any key.

NetID⁺ = { NetID, SNo,Degree, MajorDno }

SNo⁺ ={ NetID, SNo,Degree, MajorDno }

The candidate keys are NetID and SNo. **Prime:** {NetID, SNo } **Non Prime :** { Degree MajorDno }

There are no 3NF violations since no non-prime attribute is transitively dependent on the key.

Hence the table is in 3 NF.

**3.Employee ( NetID, Salary, Emp\_type)**

NetID → Salary Emp\_type

NetID⁺ ={ NetID, Salary, Emp\_type } **Prime :** {NetID} **Non Prime :** {Salary Emp\_type}

The table is in 3NF since no non-prime attribute is transitively dependent on the key.

**4.Instructor( NetID, Rank, Office, Officehrs)**

NetID → Rank Office Officehrs

NetID⁺ ={NetID, Rank, Office, Officehrs}  **Prime :** {NetID} **Non Prime :** { Rank Office Officehrs }

The table is in 3NF since no non-prime attribute is transitively dependent on the key.

**5.Department( DNo, DName, College)**

DNo → DName College

DName → DNo College

College appears only in LHS and hence does not appear in any key.

DNo⁺ ={DNo, DName, College }

DName⁺ ={DNo, DName, College }

The candidate keys are DNo and DName. **Prime :**{ DNo, DName } **Non Prime:**{College}

There are no 3NF violations since no non-prime attribute is transitively dependent on the key.

Hence the table is in 3 NF.

**6.StudentWorker( NetID, Hours, RAflag, Researchproj, TAflag, Course)**

NetID → Hours RAflag Researchproj TAflag Course

NetID⁺ ={ NetID, Hours, RAflag, Researchproj, TAflag, Course}

**Prime:**{NetID} **Non Prime:** { Hours, RAflag, Researchproj, TAflag, Course}

The table is in 3NF since no non-prime attribute is transitively dependent on the key.

**7.Facultydept( NetID, DNo)**

**8.Course( CNo, CName, Description, Level, Credits, OfferingDno)**

CNo → CName Description Level Credits OfferingDno

CName →CNo Description Level Credits OfferingDno

Description, Level, Credits, OfferingDno appear only in RHS. So, they do not appear in any key.

CNo⁺ ={ CNo, CName, Description, Level, Credits, OfferingDno }

CName⁺ ={ CNo, CName, Description, Level, Credits, OfferingDno }

The candidate keys are CNo and CName.

**Prime:**{ CNo, CName } **Non Prime:** { Description, Level, Credits, OfferingDno }

There are no 3NF violations since no non-prime attribute is transitively dependent on the key.

Hence the table is in 3 NF.

**9.Section(ClassNumber\*, CNo, SecNo, Semester, Year, FacultyNetID, LecDays,ClassRoom, InstructionMode,Totalseats, Availseats, LecStartTime,LecEndTime)**

ClassNumber → CNo SecNo Semester Year FacultyNetID LecDays ClassRoom

InstructionMode Totalseats Availseats LecStartTime LecEndTime

CNo SecNo Semester Year → ClassNumber FacultyNetID LecDays ClassRoom

InstructionMode Totalseats Availseats LecStartTime LecEndTime

FacultyNetID, LecDays,ClassRoom, InstructionMode,Totalseats, Availseats, LecStartTime,LecEndTime appear in RHS but not in LHS. Hence, they do not appear in any key.

**Potential Prime:** {ClassNumber, CNo, SecNo, Semester, Year}

ClassNumber⁺ ={ClassNumber, CNo, SecNo, Semester, Year, FacultyNetID, LecDays,ClassRoom, InstructionMode,Totalseats, Availseats, LecStartTime,LecEndTime }

CNo SecNo Semester Year ⁺ ={ClassNumber, CNo, SecNo, Semester, Year, FacultyNetID, LecDays,ClassRoom, InstructionMode,Totalseats, Availseats, LecStartTime,LecEndTime }

**Prime:**{ClassNumber, CNo SecNo Semester Year }

**Non Prime:**{ FacultyNetID, LecDays,ClassRoom, InstructionMode,Totalseats, Availseats, LecStartTime,LecEndTime }

The table is in 3NF since no non-prime attribute is transitively dependent on the key.

**10.StudentCourse (NetID, ClassNumber, Grade)**

NetID ClassNumber → Grade

Grade appears only in RHS and hence does not appear in any key.

NetID and ClassNumber appear only in LHS and is present in every key.

NetID ClassNumber⁺={NetID, ClassNumber, Grade }

NetID ClassNumber is the only candidate key. **Prime:**{NetID,ClassNumber**} Non Prime:**{Grade}

The table is in 3NF since no non-prime attribute is transitively dependent on the key.

**12.Users(NetID,Password)**

NetID→Password **Prime**:NetID **Non Prime**: Password

The table is in 3NF since no non-prime attribute is transitively dependent on the key.

**5. SQL QUERIES**

The application uses the ‘**University**’ database created in SQL Server Management Studio.

**5.1 CREATION OF TABLES AND PRIMARY KEY CONSTRAINTS**

**1. Person**

USE [University]

GO

CREATE TABLE [dbo].[Person](

[NetID] [nvarchar](12) NOT NULL,

[SSN] [int] NULL,

[FName] [nvarchar](50) NULL,

[LName] [nvarchar](50) NULL,

[Address] [nvarchar](150) NULL,

[Sex] [nchar](1) NULL,

[Phone] [bigint] NULL,

CONSTRAINT [PK\_Person] PRIMARY KEY CLUSTERED

(

[NetID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**2. Student**

CREATE TABLE [dbo].[Student](

[NetID] [nvarchar](12) NOT NULL,

[SNo] [int] NULL,

[Degree] [nvarchar](20) NULL,

[MajorDNo] [int] NULL,

CONSTRAINT [PK\_Student] PRIMARY KEY CLUSTERED

(

[NetID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**3. Employee**

CREATE TABLE [dbo].[Employee](

[NetID] [nvarchar](12) NOT NULL,

[Salary] [float] NULL,

[Emp\_type] [nvarchar](20) NULL,

CONSTRAINT [PK\_Employee] PRIMARY KEY CLUSTERED

(

[NetID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**4. Instructor**

CREATE TABLE [dbo].[Instructor](

[NetID] [nvarchar](12) NOT NULL,

[Rank] [nvarchar](30) NULL,

[Office] [nvarchar](20) NULL,

[Officehrs] [nvarchar] (200) NULL,

CONSTRAINT [PK\_Instructor] PRIMARY KEY CLUSTERED

(

[NetID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**5. Department**

CREATE TABLE [dbo].[Department](

[DNo] [int] NOT NULL,

[DName] [nvarchar](50) NULL,

[College] [nvarchar](60) NULL,

CONSTRAINT [PK\_Department] PRIMARY KEY CLUSTERED

(

[DNo] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**6. StudentWorker**

CREATE TABLE [dbo].[StudentWorker](

[NetID] [nvarchar](12) NOT NULL,

[Hours] [float] NULL,

[RAflag] [nvarchar](10) NULL,

[ResearchTopic] [nvarchar] (100) NULL,

[TAflag] [nvarchar] (10) NULL,

[CourseNo] [nvarchar] (50) NULL,

CONSTRAINT [PK\_StudentWorker] PRIMARY KEY CLUSTERED

(

[NetID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**7. Facultydept**

CREATE TABLE [dbo].[Facultydept](

[NetID] [nvarchar](12) NOT NULL,

[DNo] [int] NOT NULL,

CONSTRAINT [PK\_Facultydept] PRIMARY KEY CLUSTERED

(

[NetID],[DNo]

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**8. Course**

CREATE TABLE [dbo].[Course](

[CNo] [nvarchar] (50) NOT NULL,

[CName] [nvarchar](70) NULL,

[Description] [nvarchar](400) NULL,

[Credits] [int] NULL,

[Level] [nvarchar](15) NULL,

[OfferingDNo] [int] NULL,

CONSTRAINT [PK\_Course] PRIMARY KEY CLUSTERED

(

[CNo] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**9. Section**

CREATE TABLE [dbo].[Section](

[ClassNumber] [int] NOT NULL,

[CNo][nvarchar](50) NULL,

[SectionNo] [int] NULL,

[Semester] [nvarchar] (20) NULL,

[Year] [int] NULL,

[FacultyNetID] [nvarchar] (12) NULL,

[LectureTime] [nvarchar] (60) NULL,

[ClassRoom] [nvarchar] (20) NULL,

[InstructionMode] [nvarchar] (60) NULL,

[TotalSeats] [int] NULL,

[AvailSeats] [int] NULL,

CONSTRAINT [PK\_Section] PRIMARY KEY CLUSTERED

(

[ClassNumber] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**11. StudentCourse**

CREATE TABLE [dbo].[StudentCourse](

[NetID] [nvarchar](12) NOT NULL,

[ClassNumber] [int] NOT NULL,

[Grade] [nvarchar] (10) NULL

CONSTRAINT [PK\_StudentCourse] PRIMARY KEY CLUSTERED

(

[NetID],[ClassNumber]

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**12 .Users**

CREATE TABLE [dbo].[Users](

[NetID] [nvarchar](12) NOT NULL,

[Grade] [nvarchar] (30) NULL

CONSTRAINT [PK\_Users] PRIMARY KEY CLUSTERED

(

[NetID]

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

**5.2 CREATION OF FOREIGN KEY CONSTRAINTS**

**1. Course**

ALTER TABLE Course ADD CONSTRAINT fk\_Course

FOREIGN KEY (OfferingDNo)REFERENCES dbo.Department(DNo)

**2. FacultyDept**

ALTER TABLE dbo.Facultydept ADD CONSTRAINT fk\_Facultydept

FOREIGN KEY (NetID) REFERENCES dbo.Instructor(NetID)

ALTER TABLE dbo.Facultydept ADD CONSTRAINT fk\_Facultydept1

FOREIGN KEY (DNo) REFERENCES dbo.Department(DNo)

**4. Section**

ALTER TABLE dbo.Section ADD CONSTRAINT fk\_Section

FOREIGN KEY (CNo) REFERENCES dbo.Course(CNo)

ALTER TABLE dbo.Section ADD CONSTRAINT fk2\_Section

FOREIGN KEY (FacultyNetID) REFERENCES dbo.Instructor(NetID)

**5. Student**

ALTER TABLE dbo.Student ADD CONSTRAINT fk\_Student1

FOREIGN KEY (NetID) REFERENCES dbo.Person(NetID)

ALTER TABLE dbo.Student ADD CONSTRAINT fk\_Student

FOREIGN KEY (MajorDNo) REFERENCES dbo.Department(DNo)

**6. StudentCourse**

ALTER TABLE dbo.StudentCourse ADD CONSTRAINT fk\_StudentCourse

FOREIGN KEY (NetID) REFERENCES dbo.Student(NetID)

ALTER TABLE dbo.StudentCourse ADD CONSTRAINT fk2\_StudentCourse

FOREIGN KEY (ClassNumber) REFERENCES dbo.Section(ClassNumber)

**7. StudentWorker**

ALTER TABLE dbo.StudentWorker ADD CONSTRAINT fk\_StudentWorker

FOREIGN KEY (CourseNo) REFERENCES dbo.Course(CNo)

ALTER TABLE dbo.StudentWorker ADD CONSTRAINT fk\_StudentWorker1

FOREIGN KEY (NetID) REFERENCES dbo.Employee(NetID)

**8. Employee**

ALTER TABLE dbo.Employee ADD CONSTRAINT fk\_Employee

FOREIGN KEY (NetID) REFERENCES dbo.Person(NetID)

**9. Instructor**

ALTER TABLE dbo.Instructor ADD CONSTRAINT fk\_Instructor

FOREIGN KEY (NetID) REFERENCES dbo.Employee(NetID)

**6. DATA IN THE TABLES:**

The tables were populated using the Designer feature of Microsoft SQL Server. The following are the steps that were used to enter data into a table using Designer.

1. In the Object Explorer of SQL Server Management Studio, navigate to the table that you want to enter data and right click it.
2. Select the ‘Edit Top 200 Rows’ option.
3. The table appears on the right side of the window with editable cells.
4. Enter the values and save the table.

The following are the data entered in each of the tables:

**1. Person:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | NetID | SSN | FName | LName | Address | Sex | Phone | | axb141226 | 589112783 | Abraham | Bennet | 5000 Legacy Dr., Ste. 494,Plano, Texas 75024 | M | 9724033404 | | axh115907 | 542715923 | Anne | Hunter | 2205 Clinton Dr., Galena Park, Texas 77547 | F | 7136788600 | | bxg171332 | 112873840 | Burt | Gringlesby | 1109 Fairmont Parkway,Pasadena, Texas 77504 | M | 2814853117 | | cxc134957 | 261830355 | Cheryl | Carson | 329 Neches Street,Jacksonville, Texas 75766 | F | 9365644252 | | cxl129847 | 312456896 | Charlene | Locksley | 7 North Park Plaza,Brownsville, Texas 78521 | F | 9565480227 | | dxs209865 | 709218326 | Dean | Straight | 5206 Irvington Blvd., Unit D,Houston, Texas 77009 | M | 7137425000 | | jxn121387 | 845527317 | Jane | Nelson | 803 Yale Street ,Houston, TX 77007 | F | 8234576198 | | jxs179854 | NULL | Jim | Smith | 11451 Katy Freeway, Suite 209 ,Houston, Texas 77079 | M | 5124630106 | | jxw190459 | 435986125 | Johnson | White | 2525 Kell Blvd., Suite 302,Wichita Falls, Texas 76308 | M | 4087693458 | | mxg113476 | 987456321 | Margie | Green | 300 W. Schuster Ave.,El Paso, TX 79902 | F | 9154749870 | | pxv120830 | NULL | Praveen | Venkat | 2915 Atkinson Drive,Lufkin, Texas 75901 | M | 3699998955 | | rxk120841 | 812340214 | Ram | Kumar | 5411 Plaza Drive Suite D,Texarkana, Texas 75503 | M | 9032237931 | | rxm121331 | NULL | Revathi | Murthi | 5787 S HAMPTON RD STE 130,DALLAS TX 75232-2289 | F | 9123847560 | | rxn121330 | NULL | Rekha Muthulakshmi | Nachadalingam | 7740 McCallum Blvd,Apt 162,Dallas,TX 75252 | F | 2148087643 | | rxr301293 | 873482013 | Raji | Ramodharan | 925 North Frost Center,1250 N.E. Loop 410,San Antonio, Texas 78209 | F | 2135635633 | | sxr1143589 | NULL | Saranya | Radhakrishnan | 1235 S. Main Street, Suite 280,Grapevine, Texas 76051 | F | 8174886648 | | sxs126345 | 543612775 | Swetha | Saravanan | 5847 San Felipe, Suite 2400,Houston, Texas 77057 | F | 2134255678 | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **2.Student**   |  |  |  |  | | --- | --- | --- | --- | | NetID | SNo | Degree | MajorDNo | | pxv120830 | 2021085675 | Master of Science in Supply Chain Management | 3 | | rxm121331 | 2021832879 | Master of Science in Computer Engineering | 1 | | rxn121330 | 2021125978 | Master of Science in Computer Science | 1 | | sxr1143589 | 2021114679 | Bachelor of Science in Interdisciplinary Studies | 9 | | sxs126345 | 2020349844 | Master of Science in Computer Science | 1 | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **3. Employee**   |  |  |  | | --- | --- | --- | | NetID | Salary | Emp\_type | | axb141226 | 7000 | Instructor | | axh115907 | 8000 | Instructor | | bxg171332 | 2000 | Student Worker | | cxc134957 | 6000 | Instructor | | cxl129847 | 6600 | Instructor | | dxs209865 | 8200 | Instructor | | jxn121387 | 1200 | Student Worker | | jxs179854 | 7400 | Instructor | | jxw190459 | 8500 | Instructor | | mxg113476 | 1000 | Student Worker | | rxk120841 | 7500 | Instructor | | rxr301293 | 5000 | Instructor | |  |  |  |  |  |  |
| **4. Instructor**   |  |  |  |  | | --- | --- | --- | --- | | NetID | Rank | Office | Officehrs | | axb141226 | Professor | ECSS 4.12 | Mon,Wed 1:00 pm-3:00 pm | | axh115907 | Professor | ESCN 3.11 | Tues 10:00 am- 1:00 pm | | cxc134957 | Assistant Professor | JSOM 2.15 | Thurs,Fri 2:00 pm-3:00 pm | | cxl129847 | Professor | FO 2.13 | Thurs,Fri 1:00 pm-3:30 pm | | jxs179854 | Assoc. Professor | MECH 3.09 | Mon 9:00 am-11:00 am | | jxw190459 | Department Head | ECSS 3.16 | Wed,Fri 1:00 pm-4:00 pm | | rxk120841 | Assoc. Professor | JSOM 1.13 | Tues,Thurs 5:00 pm-7:00 pm | | rxr301293 | Assistant Professor | ATEC 4.112 | Fri 11:00 am-3:00 pm |   **5. Department**   |  |  |  | | --- | --- | --- | | DNo | DName | College | | 1 | Computer Science | Eric Jonsson School of Computer Science | | 2 | Electrical Engineering | Eric Jonsson School of Engineering | | 3 | Information Technology and Management | Jindal School of Management | | 4 | GeoInformatics | School of Natural Sciences and Mathematics | | 5 | Mechanical Engineering | Eric Jonsson School of Engineering | | 6 | Finance and Managerial Economics | Jindal School of Management | | 7 | Arts and Technology | School of Arts and Humanities | | 8 | Chemistry | School of Natural Sciences and Mathematics | | 9 | Environment Studies | School of Interdisciplinary Studies | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **6. StudentWorker**   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | NetID | Hours | RAflag | ResearchTopic | TAflag | CourseNo | | bxg171332 | 20 | 0 | NULL | 1 | 6301 | | jxn121387 | 15 | 1 | Computational logic and intelligent computer systems | 0 | NULL | | mxg113476 | 17 | 0 | NULL | 1 | 2310 | |  |  |  |  |  |  |
| **7.Facultydept**   |  |  | | --- | --- | | NetID | DNo | | axb141226 | 2 | | axb141226 | 6 | | axh115907 | 9 | | cxc134957 | 3 | | cxl129847 | 1 | | jxs179854 | 5 | | jxw190459 | 1 | | rxk120841 | 3 | | rxr301293 | 7 | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **8.Course**   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | CNo | CName | Description | Credits | Level | OfferingDNo | | ATEC 2382 | Computer Imaging | Introduction to digital image-making and manipulation using contemporary software applications. Graphic and photographic methods are presented and applied to art and design problems. Computer images are prepared for multiple delivery environments, including the Internet, games, animation, and print. | 3 | Undergraduate | 7 | | BIS 3190 | Library Research Skills | Through this course, students will develop the skills to identify what information is needed for their research, how to find and evaluate scholarly resources, and how to organize the information for a paper or other course project. Plagiarism, copyright, and citation management will be included | 1 | Undergraduate | 9 | | BPS 6310 | Strategic Management | Strategic management consists of the analysis, decisions, and actions that organizations take to create sustainable competitive advantages. The course examines a variety of issues including environmental, competitor, and stakeholder analysis; strategy formulation; and strategy implementation and control. | 3 | Graduate | 3 | | CS 5330 | Computer Science II | NULL | 3 | Graduate | 1 | | CS 5343 | Algorithm Analysis & Data Structures | Formal specifications and representation of lists, arrays, trees, graphs, multilinked structures, strings and recursive pattern structures. Analysis of associated algorithms. Sorting and searching, file structures. Relational data models. | 3 | Graduate | 1 | | CS 5348 | Operating Systems Concepts | Processes and threads. Concurrency issues including semaphores, monitors and deadlocks. Simple memory management. Virtual memory management. CPU scheduling algorithms. I/O management. File management. Introduction to distributed systems. | 3 | Graduate | 1 | | EE 2310 | Introduction to Digital Systems | Introduction to digital circuits, hardware structures, and assembly-language concepts that underlie the design of modern computer systems. Topics include: Internal data representation and arithmetic operations in a computer, basic logic circuits | 3 | Undergraduate | 2 | | MKT 6301 | Marketing Management | Overview of marketing management methods, principles and concepts including product, pricing, promotion and distribution decisions as well as segmentation, targeting and positioning. | 3 | Graduate | 3 | | OB 6301 | Organization Behavior | The study of human behavior in organizations. Emphasizes theoretical concepts and practical methods for understanding, analyzing, and predicting individual, group, and organizational behavior. Topics include work motivation, group dynamics, decision making, conflict and negotiation, leadership, power | 3 | Graduate | 3 | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **9.Section**   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | ClassNumber | CNo | SectionNo | Semester | Year | FacultyNetID | LectureDays | | 10468 | CS 5330 | 201 | Spring | 2013 | jxw190459 | Tuesday & Thursday | | 13437 | MKT 6301 | 501 | Fall | 2012 | axb141226 | Tuesday | | 13726 | BPS 6310 | 502 | Fall | 2012 | cxc134957 | Thursday | | 21180 | EE 2310 | 501 | Fall | 2012 | axb141226 | Tuesday & Thursday | | 21267 | ATEC 2382 | 501 | Spring | 2013 | rxr301293 | Monday | | 22262 | CS 5348 | 501 | Spring | 2013 | cxl129847 | Tuesday & Thursday | | 22266 | CS 5343 | 502 | Spring | 2013 | jxw190459 | Tuesday & Thursday | | 22268 | CS 5343 | 501 | Spring | 2013 | jxw190459 | Friday | | 23443 | OB 6301 | 500 | Fall | 2012 | rxk120841 | Friday | | 24028 | BIS 3190 | 202 | Fall | 2012 | axh115907 | Monday & Wednesday |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | ClassRoom | InstructionMode | TotalSeats | AvailSeats | LecStartTime | LecEndTime | | GR 2.530 | Face-to-Face | 5 | 5 | 5:30 PM | 6:45 PM | | JSOM 2.717 | Face-to-Face | 75 | 75 | 7:00 PM | 9:45 PM | | JSOM 2.902 | Face-to-Face | 50 | 50 | 8:00 AM | 10:45 AM | | ECSS 2.415 | Face-to-Face | 30 | 30 | 11:30 AM | 12.45 PM | | ATEC 1.102 | Face-to-Face | 20 | 20 | 10:00 AM | 12.45 PM | | FO 1.502 | Face-to-Face | 15 | 15 | 10:00 AM | 11:15 AM | | FO 1.202 | Face-to-Face | 40 | 40 | 2:30 PM | 3:45 PM | | FO 2.102 | Face-to-Face | 5 | 5 | 4:00 PM | 6:45 PM | | JSOM 2.722 | Face-to-Face | 60 | 60 | 7:00 PM | 9:45 PM | | NULL | Online | 20 | 20 | 4:00 PM | 5:15 PM | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| 10.StudentCourse   |  |  |  | | --- | --- | --- | | NetID | ClassNumber | Grade | | rxn121330 | 10468 | A | | rxn121330 | 24028 | A- | | sxr1143589 | 24028 | B | | sxs126345 | 10468 | A |   **7. STORED PROCEDURES CREATED TO IMPLEMENT THE SYSTEM** |  |  |  |  |  |  |

**1. sp\_Enroll**

CREATE PROCEDURE [dbo].[sp\_Enroll]

@Studentid nvarchar(20),

@ClassNumber int,

@ReturnVal INT =NULL OUTPUT

AS

BEGIN

SET NOCOUNT ON;

DECLARE @avail INT

IF EXISTS(SELECT \* FROM Student WHERE NetID=@Studentid)

BEGIN

IF EXISTS(SELECT \* FROM Section WHERE ClassNumber=@ClassNumber)

BEGIN

SELECT @avail=AvailSeats FROM Section WHERE ClassNumber=@ClassNumber

IF NOT EXISTS(SELECT \* FROM StudentCourse WHERE ClassNumber=@ClassNumber AND NetID=@Studentid)

BEGIN

IF @avail<=0 --No Seats Available

SET @ReturnVal= 0

ELSE

BEGIN

INSERT INTO StudentCourse VALUES(@Studentid,@ClassNumber,NULL)

UPDATE Section SET AvailSeats=AvailSeats-1 WHERE ClassNumber=@ClassNumber

set @ReturnVal=1 --Successfully Enrolled

END

END

ELSE

set @ReturnVal=2 --Student is already enrolled

END

ELSE

set @ReturnVal=3 ---Invalid Class Number

END

ELSE

SET @ReturnVal=4 ---Invalid Student ID

RETURN @ReturnVal

END

**2. sp\_Drop**

CREATE PROCEDURE [dbo].[sp\_Drop]

@Studentid nvarchar(20),

@ClassNumber int,

@ReturnVal int= NULL OUTPUT

AS

BEGIN

SET NOCOUNT ON;

IF EXISTS(SELECT \* FROM Student WHERE NetID=@Studentid)

BEGIN

IF EXISTS(SELECT \* FROM Section WHERE ClassNumber=@ClassNumber)

BEGIN

IF EXISTS(SELECT \* FROM StudentCourse WHERE NetID=@Studentid AND ClassNumber=@ClassNumber)

BEGIN

DELETE FROM StudentCourse WHERE NetID=@Studentid AND ClassNumber=@ClassNumber

UPDATE Section SET AvailSeats=AvailSeats+1 WHERE ClassNumber=@ClassNumber

SET @ReturnVal=1 ---Dropped Successfully

END

ELSE

SET @ReturnVal=0

END

ELSE

SET @ReturnVal=2 ---Invalid Class Number

END

ELSE

SET @ReturnVal=3 --Invalid Student ID

RETURN @ReturnVal

END

**3. sp\_ClassSchedule**

CREATE PROCEDURE [dbo].[sp\_ClassSchedule]

@netid nvarchar(20)

AS

BEGIN

SET NOCOUNT ON;

SELECT C.CNo,S.SectionNo as SecNo,S.ClassNumber as ClassNo,S.LectureDays,S.LecStartTime,S.LecEndTime,S.ClassRoom

FROM Course C, Section S,StudentCourse sc

WHERE C.CNo=S.CNo AND sc.NetID=@netid AND sc.ClassNumber=s.ClassNumber

END

**4. sp\_CourseLookup**

CREATE PROCEDURE [dbo].[sp\_CourseLookup]

@CName NVARCHAR(70)=NULL,

@CNo nvarchar(50)=NULL,

@Semester nvarchar(20)=NULL,

@Year int=NULL,

@Instructor nvarchar(50)=NULL,

@LecStartTime nvarchar(20)=NULL,

@LecEndTime nvarchar(20)=NULL

AS

BEGIN

SET NOCOUNT ON;

SELECT S.ClassNumber,S.CNo,C.CName,S.SectionNo,S.Semester,S.Year,S.LectureDays ,S.LecStartTime,S.LecEndTime,P.FName,P.LName

FROM Course C, Section S,Person P

WHERE

S.CNo=C.CNo AND S.FacultyNetID=P.NetID

AND

(S.CNo=@CNo OR @CNo IS NULL)

AND

(C.CName=@CName OR @CName IS NULL)

AND

(S.Semester=@Semester OR @Semester IS NULL)

AND

(S.Year=@Year OR @Year IS NULL)

AND

(P.LName=@Instructor OR @Instructor IS NULL)

AND

(S.LecStartTime=@LecStartTime OR @LecStartTime IS NULL)

AND

(S.LecEndTime=@LecEndTime OR @LecEndTime IS NULL)

END

**5. sp\_FacultyDetails**

CREATE PROCEDURE [dbo].[sp\_FacultyDetails]

@lname nvarchar(50)

AS

BEGIN

SET NOCOUNT ON;

SELECT dbo.Instructor.NetID,dbo.Person.FName, dbo.Person.LName, dbo.Instructor.Rank,dbo.Instructor.Office, dbo.Instructor.Officehrs

FROM dbo.Instructor,dbo.Person

WHERE dbo.Instructor.NetID = dbo.Person.NetID

AND dbo.Person.LName like @lname+'%'

END

**6. sp\_LoadData**

CREATE PROCEDURE [dbo].[sp\_LoadData]

AS

BEGIN

SET NOCOUNT ON;

SELECT DISTINCT CName FROM Course

SELECT DISTINCT CNo FROM Course

SELECT P.FName,P.LName FROM Person P

WHERE

P.NetID IN (SELECT NetID FROM Instructor)

SELECT DISTINCT LecStartTime FROM Section

SELECT DISTINCT LecEndTime FROM Section

END

**7. sp\_PersonalInfo**

CREATE PROCEDURE [dbo].[sp\_PersonalInfo]

@NetID nvarchar(20)

AS

BEGIN

SET NOCOUNT ON;

SELECT s.SNo as StudentNo,P.FName,P.LName,P.Address,P.Phone,P.Sex,S.Degree

FROM Person P,Student S

WHERE P.NetID=@NetID AND S.NetID=P.NetID

END

**8.sp\_UpdatePersonalInfo**

CREATE PROCEDURE [dbo].[sp\_UpdatePersonalInfo]

@Studentid nvarchar(20),

@Fname nvarchar(50),

@Lname nvarchar(50),

@Address nvarchar(150),

@Phone bigint,

@Sex nchar(1),

@Degree nvarchar(70)

AS

BEGIN

SET NOCOUNT ON;

UPDATE Person

SET FName=@Fname,LName=@Lname,Address=@Address,Phone=@Phone,Sex=@Sex

WHERE NetID=@Studentid

UPDATE Student

SET Degree=@Degree

WHERE NetID=@Studentid

END

**9. sp\_StudentProfile**

CREATE PROCEDURE [dbo].[sp\_StudentProfile]

@NetID nvarchar(20)

AS

BEGIN

SET NOCOUNT ON;

SELECT s.CNo,c.CName,sc.ClassNumber as ClassNo,sc.Grade

FROM

StudentCourse sc,Section s,Course c

WHERE

sc.ClassNumber=s.ClassNumber AND s.CNo=c.CNo AND sc.NetID=@NetID

END

**10. sp\_ViewClassDetail**

CREATE PROCEDURE [dbo].[sp\_ViewClassDetail]

@ClassNumber int

AS

BEGIN

SET NOCOUNT ON;

SELECT CNo,SectionNo,Semester,Year,FacultyNetID,LectureDays,LecStartTime,

LecEndTime,TotalSeats,AvailSeats

FROM Section

WHERE ClassNumber=@ClassNumber

END

**11. sp\_Students\_List**

CREATE PROCEDURE [dbo].[Students\_List]

@ClassNumber int,

@ReturnVal INT =NULL OUTPUT

AS

BEGIN

SET NOCOUNT ON;

IF EXISTS(SELECT \* FROM Section WHERE ClassNumber=@ClassNumber)

BEGIN

SELECT S.NetID,FName,LName

FROM Person P, StudentCourse S

WHERE S.NetID=P.NetID AND S.ClassNumber=@ClassNumber

set @ReturnVal=1

END

ELSE

set @ReturnVal=2

return @ReturnVal

END

**12. sp\_login**

CREATE PROCEDURE [dbo].[sp\_login]

@UserName nvarchar(20),

@Password nvarchar(30)

AS

BEGIN

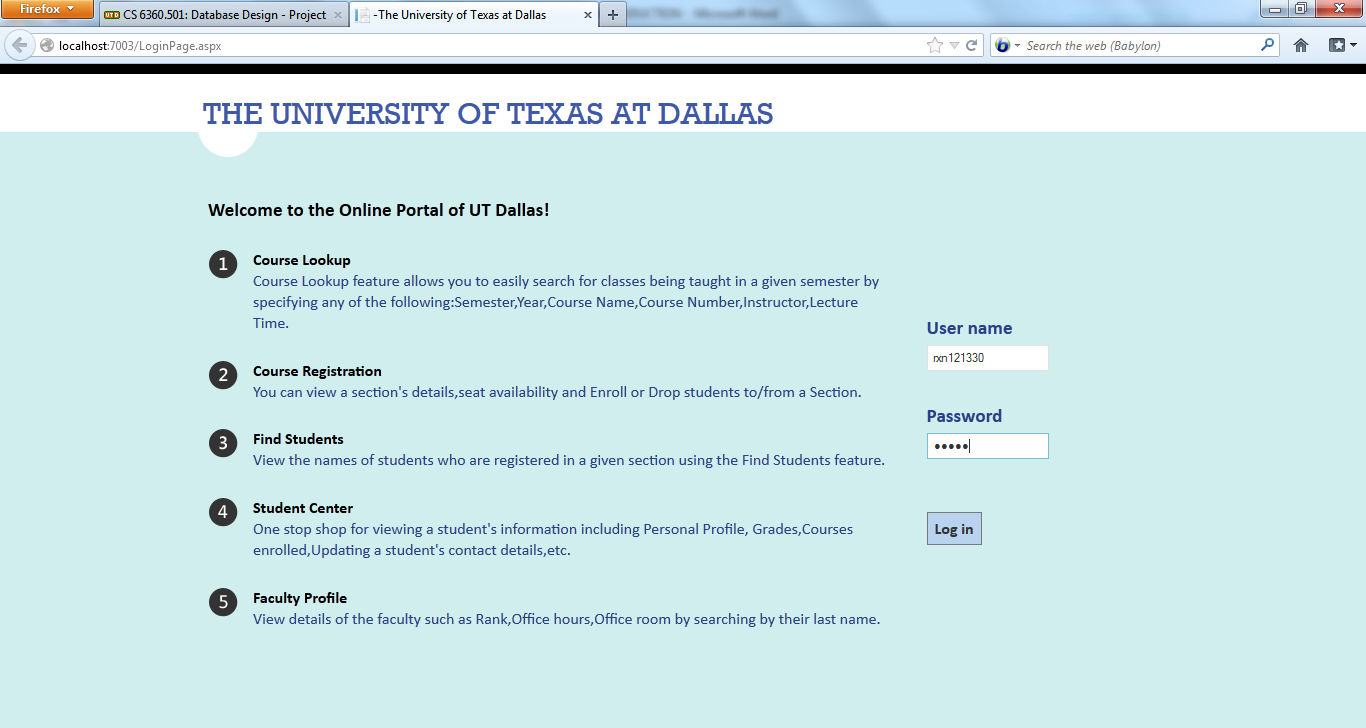
SET NOCOUNT ON;

SELECT \* FROM Users WHERE NetID=@UserName

END

**8. DEMONSTRATION OF THE APPLICATION**

**8.1. Login Page**

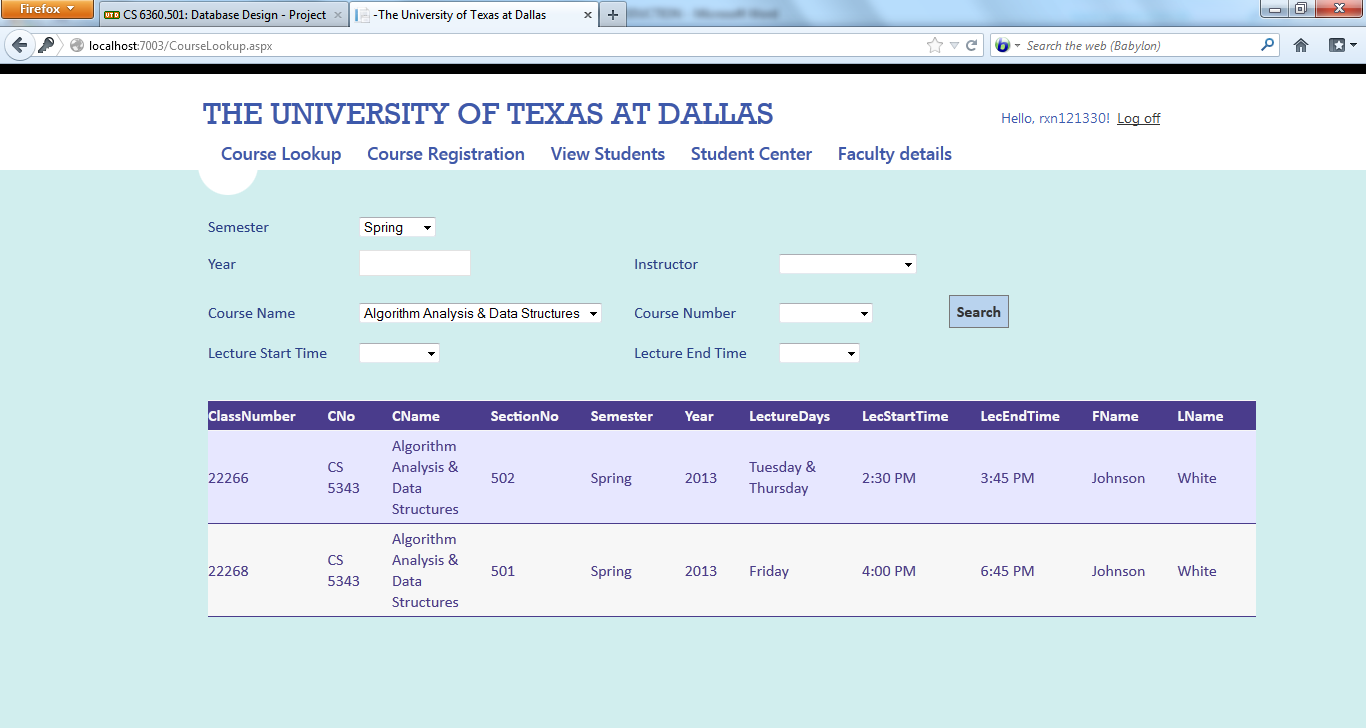
****

**8.2. Course Lookup**

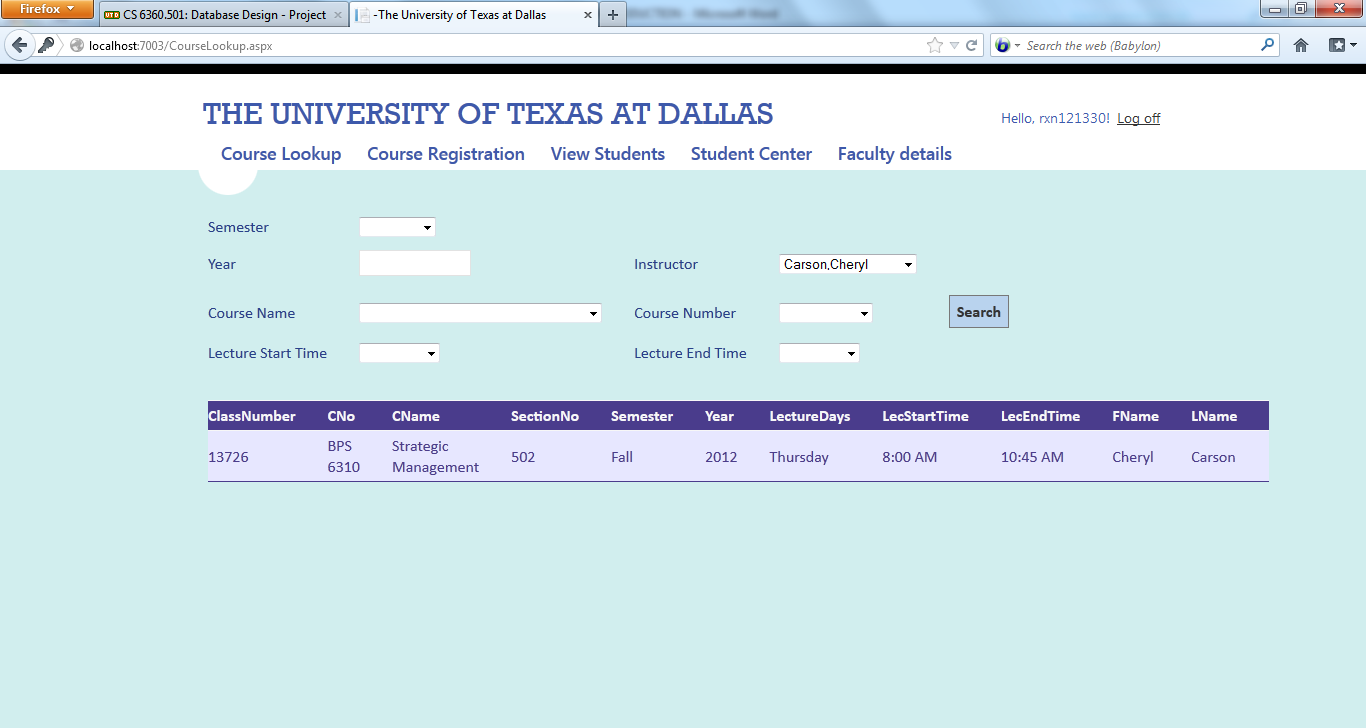
**8.2.1 Search by Semester**

****

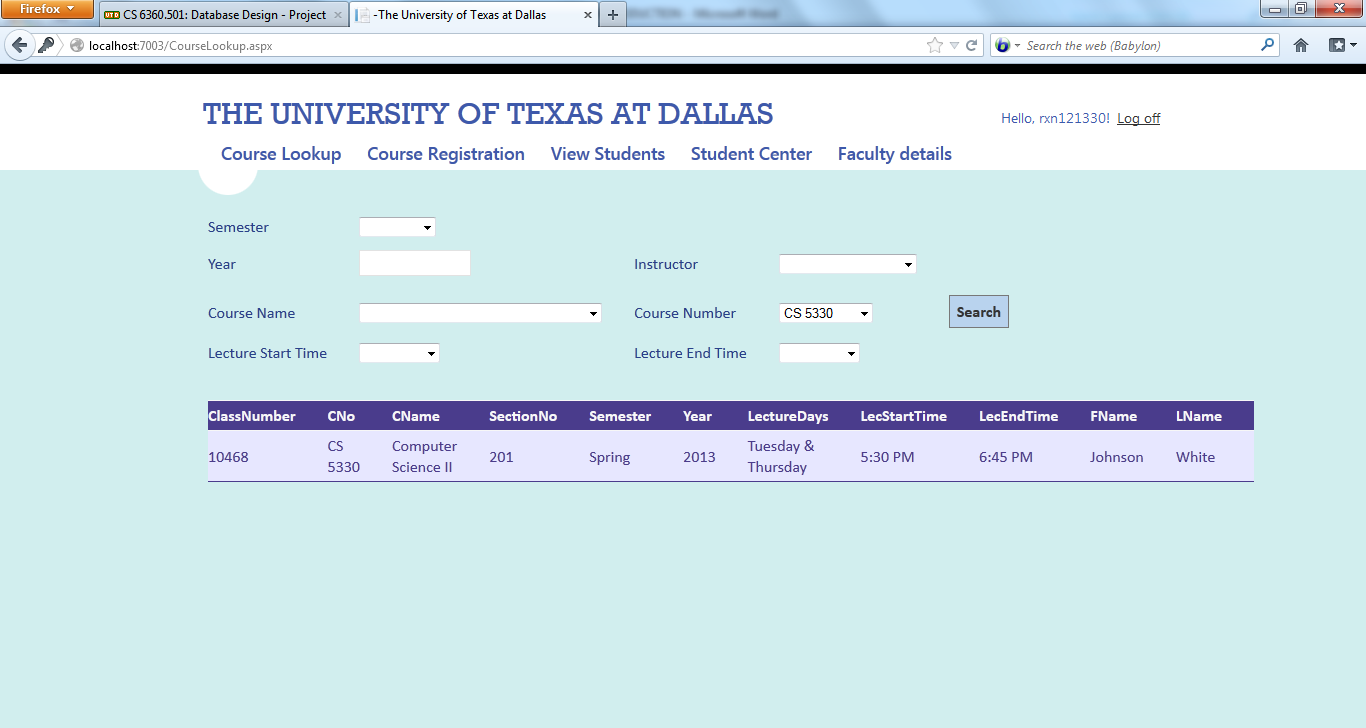
**8.2.2 Search by Course Name**

****

**8.2.3 Search by Instructor**



**8.2.4 Search by Course Number**

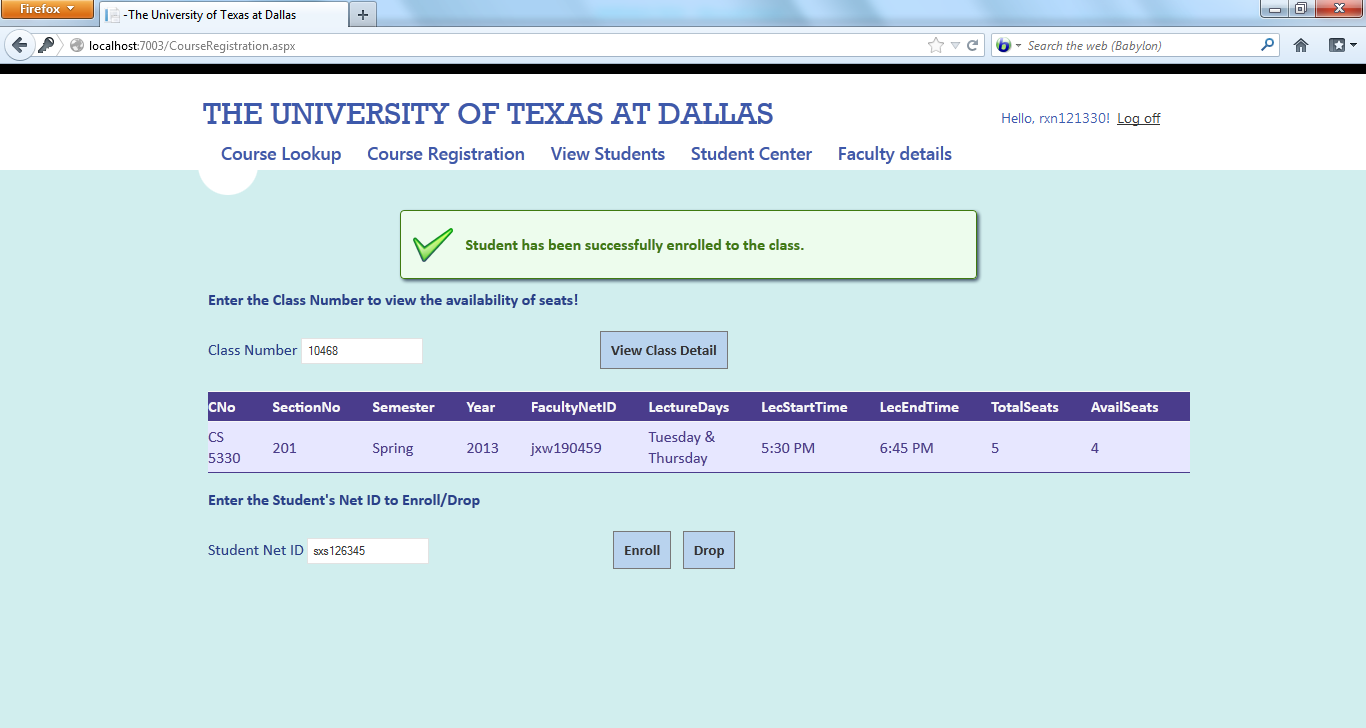


**8.2.5** **Search by Lecture Time**

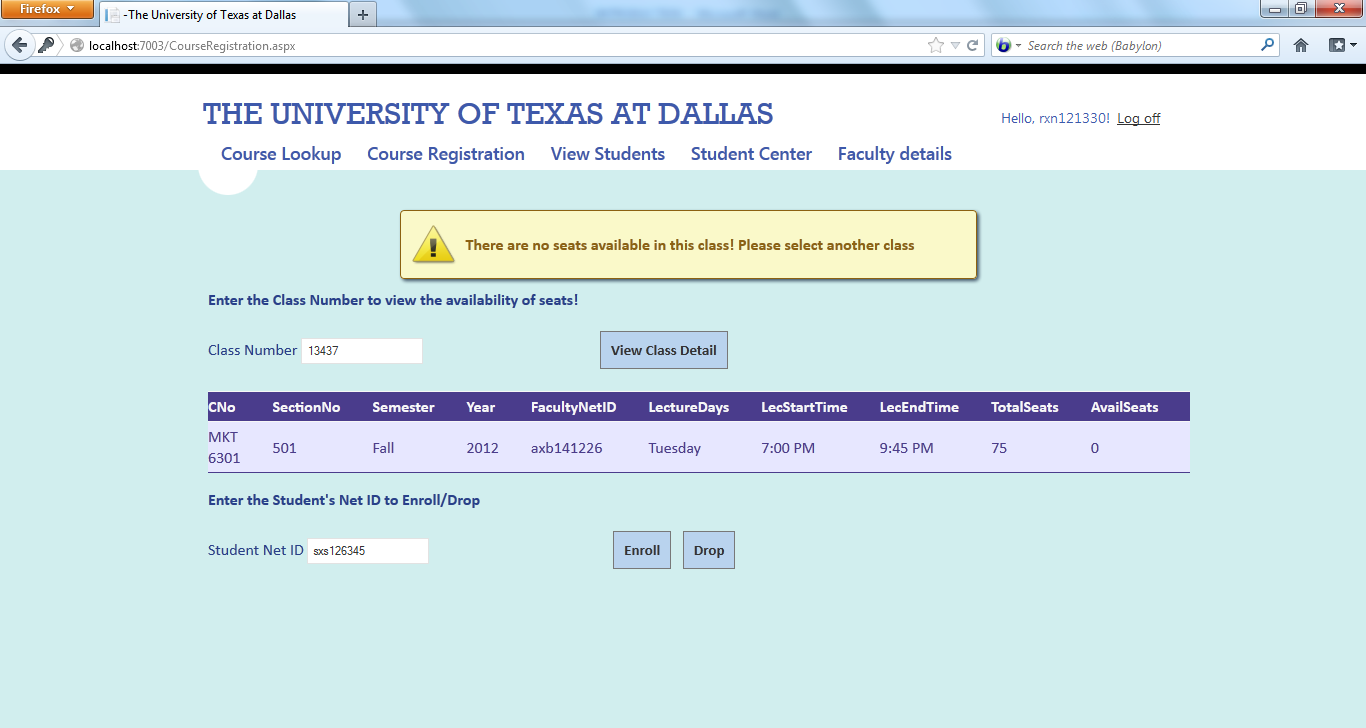
****

**8.3. Course Registration**

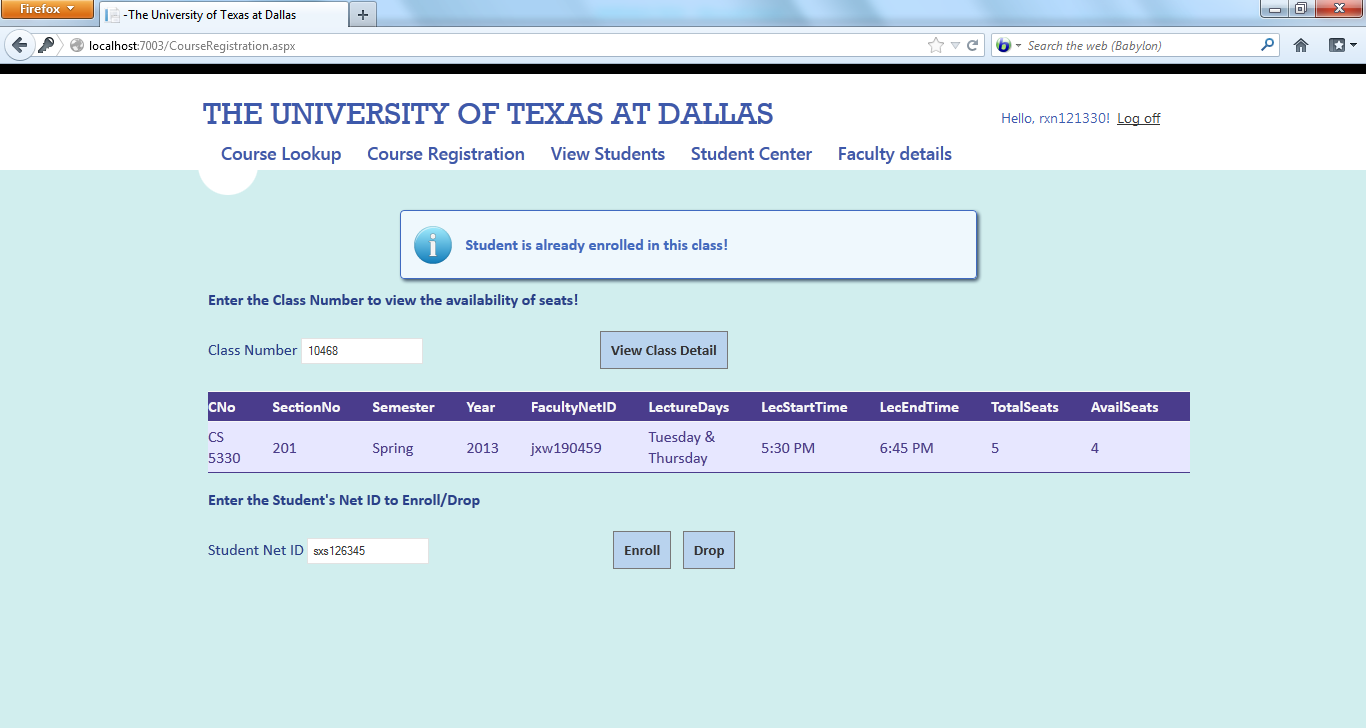
**8.3.1 Successful enrollment**

****

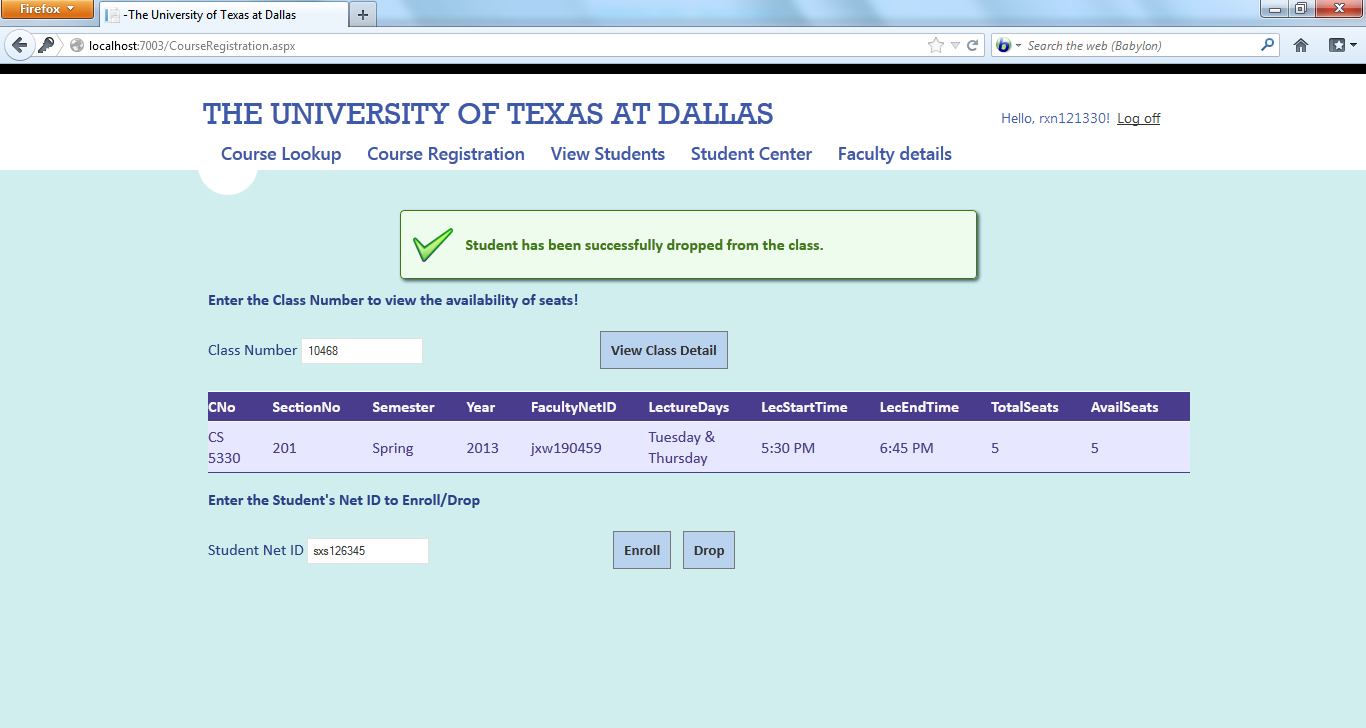
**8.3.2 No seats available**

****

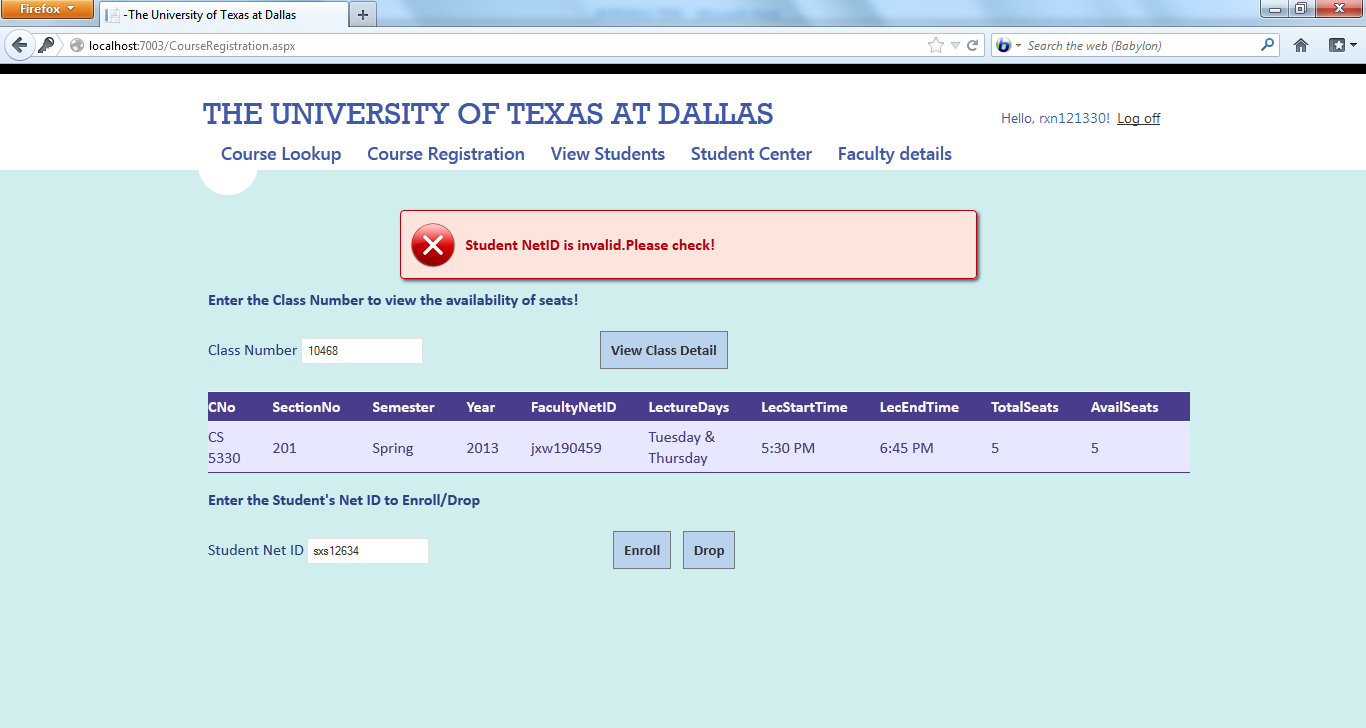
**8.3.3 Student is already enrolled**



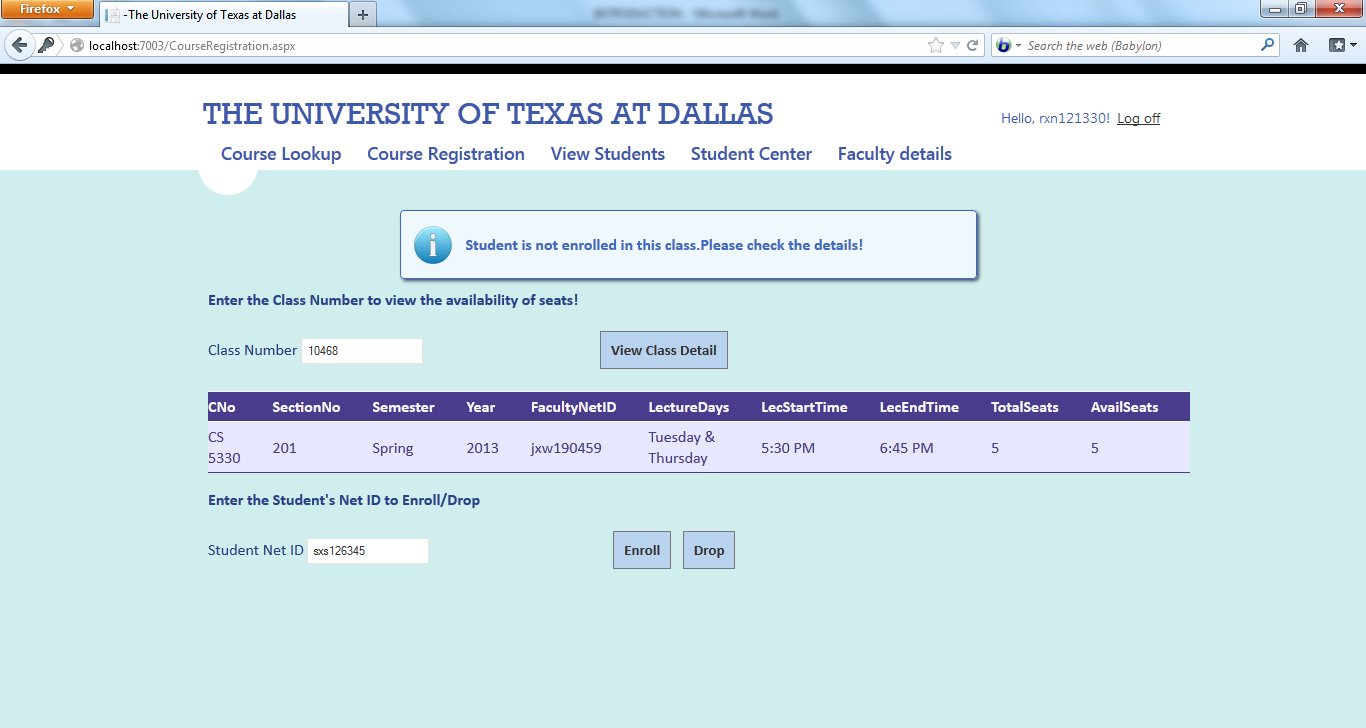
**8.4 Drop from Course**



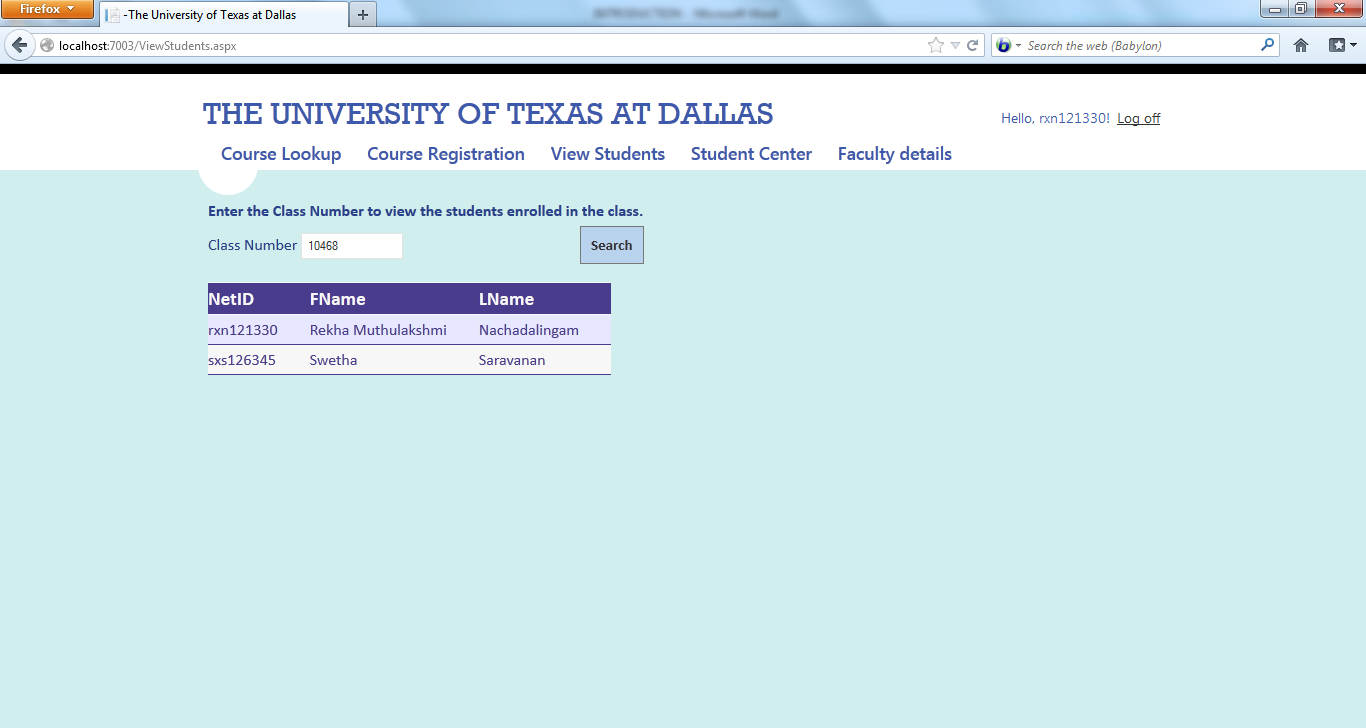
**8.4.1 Invalid Student ID**



**8.4.2 Student does not belong to that class**



**8.5 View Students enrolled in a section**



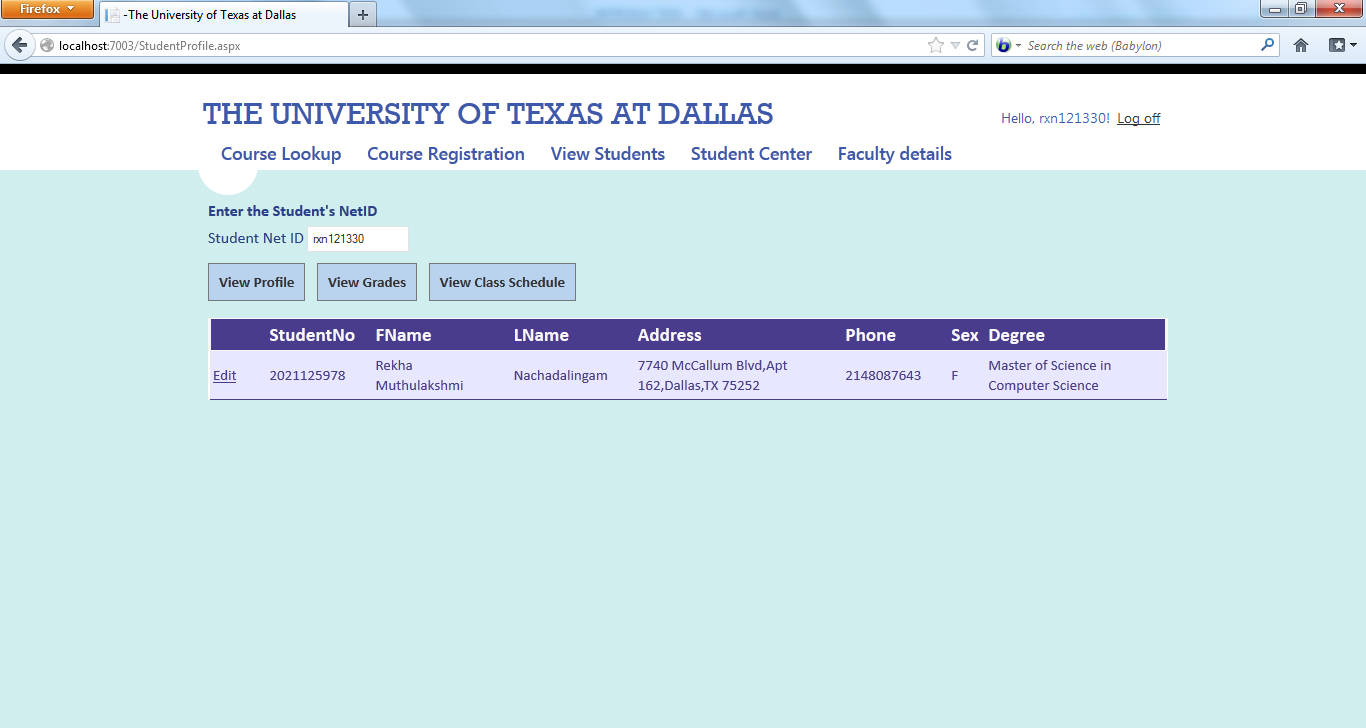
**9. ADDITIONAL FUNCTIONALITY IMPLEMENTED**

**9.1. View courses and grades of a student**

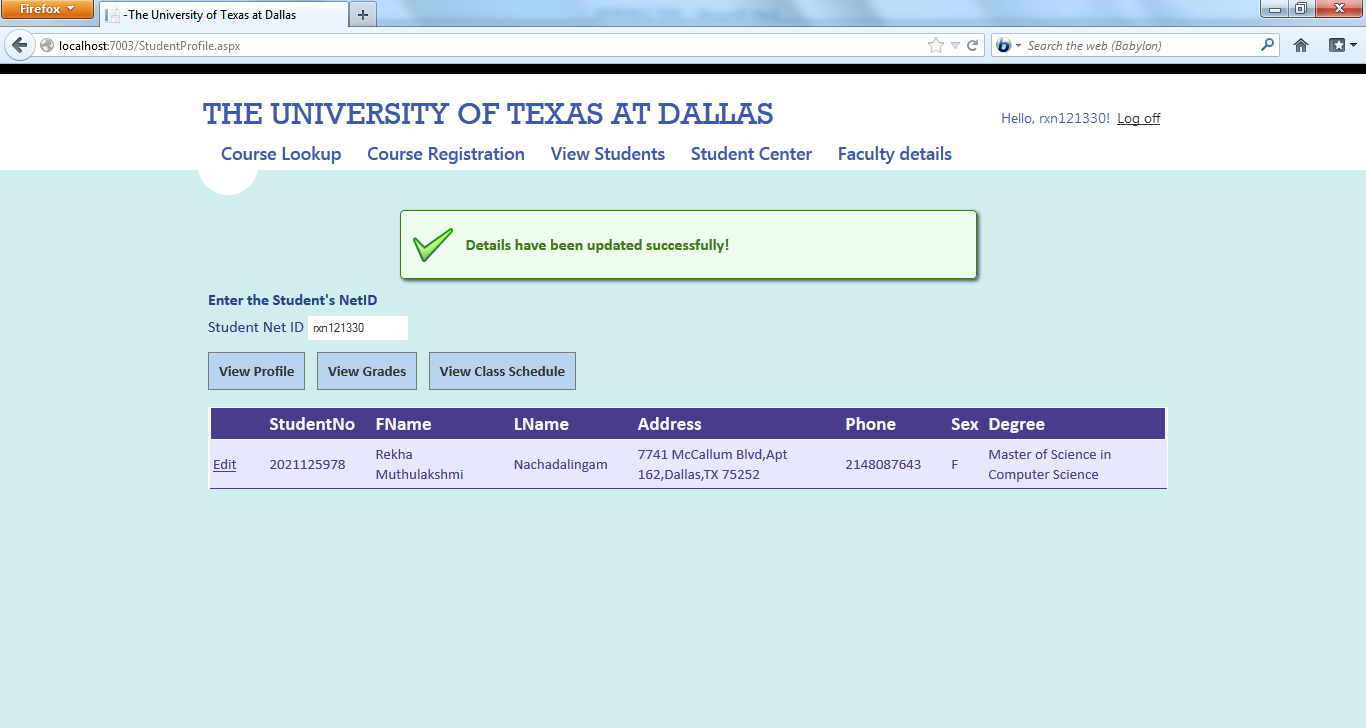


**9.2. View and edit student’s personal details**

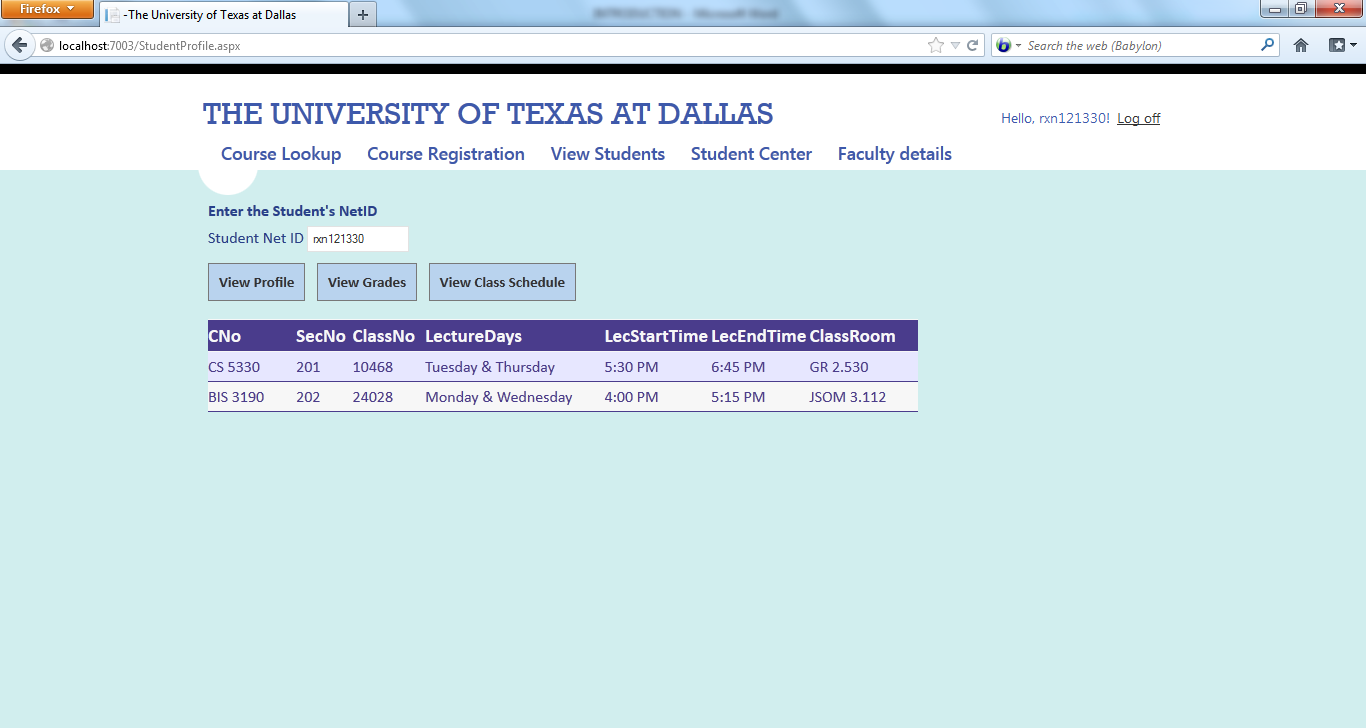
**View details:**



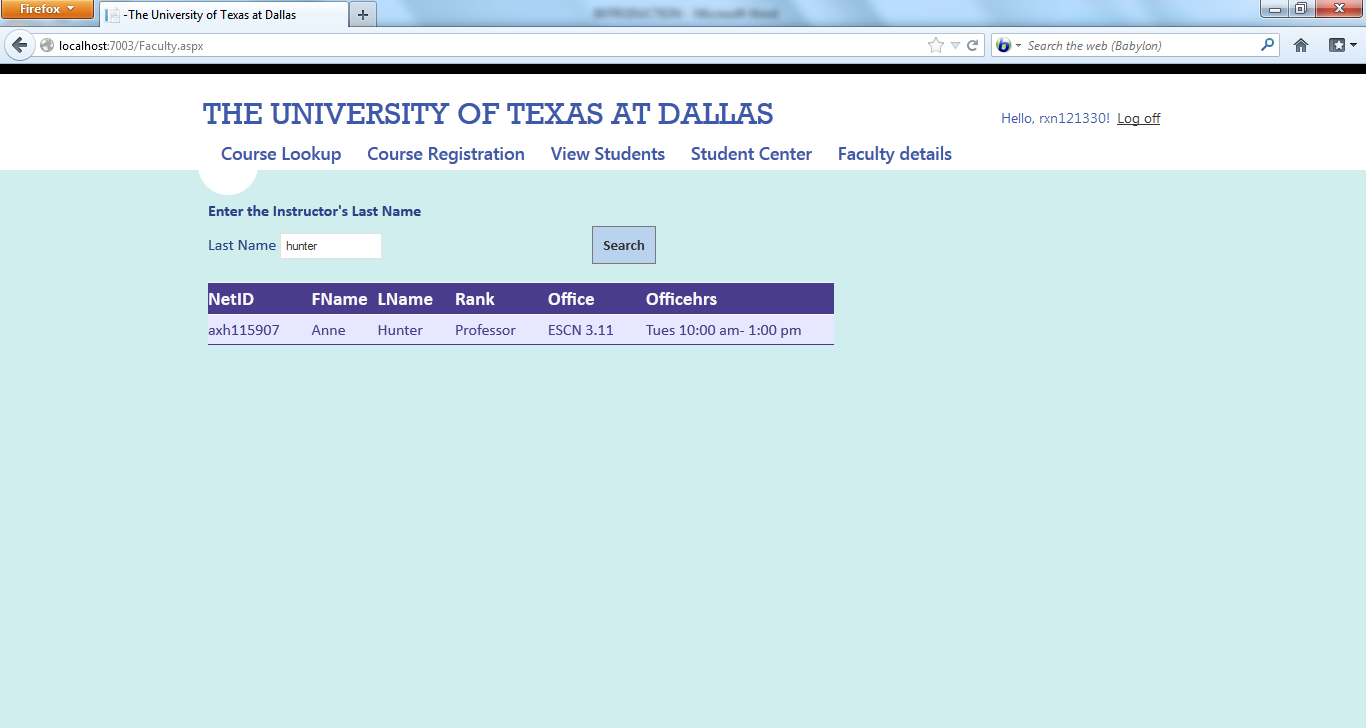
**Edit Details:**



**9.3. View Class schedule of a student**



**9. 5. View faculty details**



**10. CONCLUSION**

The following are the plus features and limitations of the application.

**PLUS:**

* The application is free from SQL Injection attacks since all the features of the application have been implemented using Stored Procedures.
* The application has a good error handling mechanism and all the User and System Exceptions are caught and handled appropriately.
* The application maintains a consistent look and feel in all its pages and provides easy navigation through the use of menus.

**LIMITATIONS:**

* The application is designed such all the pages are accessible to the students and faculty of the University. Access can be further restricted by providing roles and access privileges to the users.

**11. REFERENCES**

1. <http://www.asp.net/web-forms>

2. <http://www.nakov.com>

3. <http://msdn.microsoft.com>