

Information System for Course Assignments

Ben Albot Diego Lazo Rex Keen Eric Sanchez Tara Parker

April 28, 2014

Abstract

We attempt to design and develop an information system to record which instructor teaches which course(s) in a given semester with what resources, and to enable users to answer questions about the instructors and courses etc. Our aim is to develop a high quality design and deliver a system meeting the project description such that end users can use it, and the system will remain as a demonstration for future students of the Concepts of Databases Systems course.

Contents

1	UML Data Model	5
2	Relation Schemas	7
2.1	Courses	7
2.1.1	courseCode	7
2.1.2	catalogYear	7
2.1.3	courseTitle	7
2.1.4	required	8
2.2	Sections	8
2.2.1	CRN	8
2.2.2	year	8
2.2.3	semester	8
2.2.4	sectionNumber	8
2.2.5	type	8
2.2.6	days	8
2.2.7	startTime	9
2.2.8	endTime	9
2.2.9	enrollment	9
2.2.10	capacity	9
2.2.11	room	9
2.2.12	bldg	9
2.2.13	descriptions	9
2.3	Instructors	9
2.3.1	rNumber	10
2.3.2	lastName	10
2.3.3	firstName	10
2.3.4	instructorTitle	10
2.3.5	tenured	10

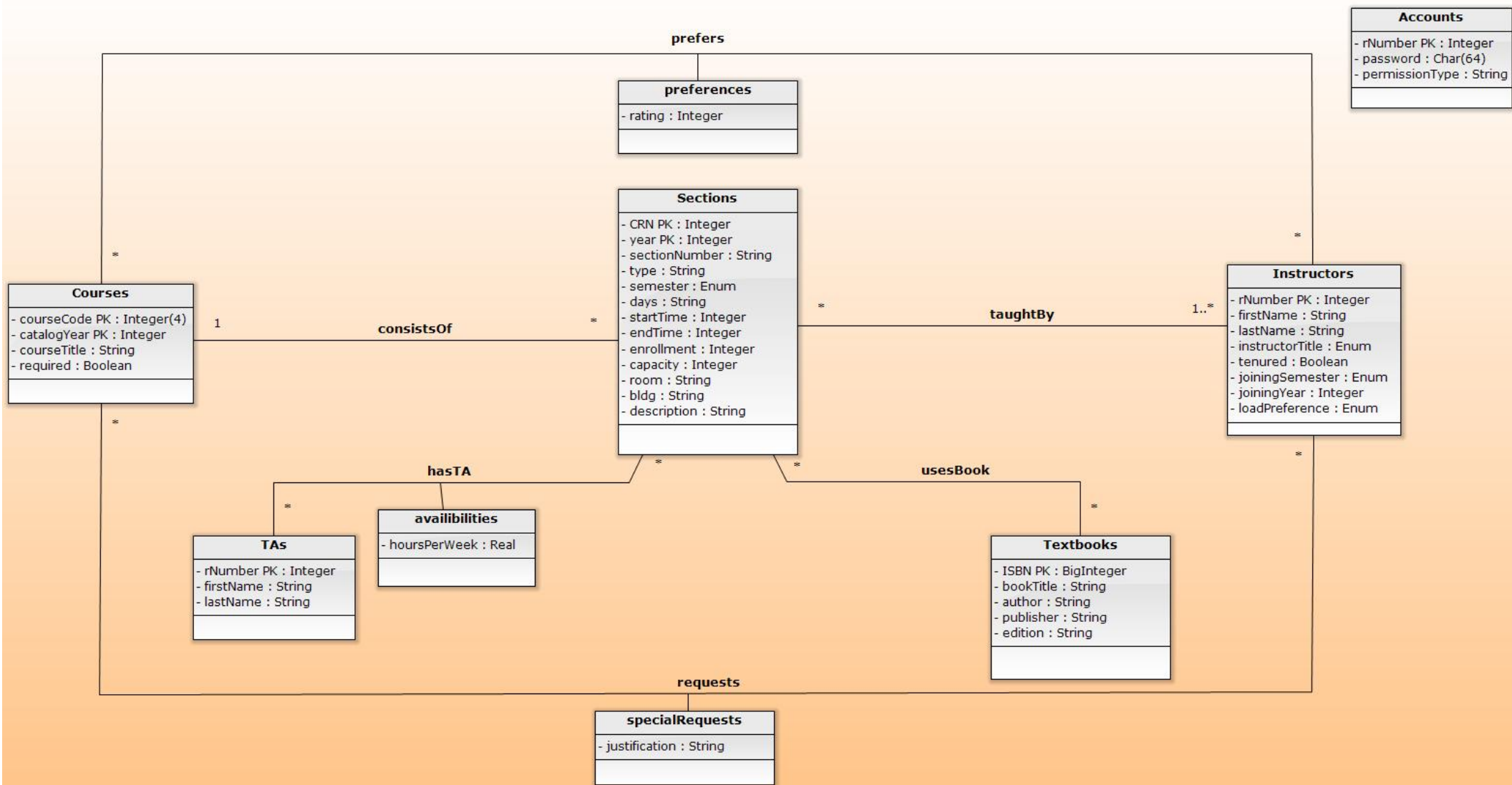
2.3.6	joiningSemester	10
2.3.7	joiningYear	10
2.3.8	loadPreference	10
2.4	TAs	11
2.4.1	rNumber	11
2.4.2	lastName	11
2.4.3	firstName	11
2.5	Textbooks	11
2.5.1	ISBN	11
2.5.2	bookTitle	11
2.5.3	author	11
2.5.4	publisher	11
2.5.5	edition	12
2.6	consistsOf	12
2.6.1	CRN	12
2.6.2	semester	12
2.6.3	year	12
2.6.4	courseCode	12
2.6.5	catalogYear	12
2.7	taughtBy	13
2.7.1	CRN	13
2.7.2	semester	13
2.7.3	year	13
2.7.4	rNumber	13
2.8	HasTA	13
2.8.1	CRN	13
2.8.2	semester	13
2.8.3	year	14
2.8.4	rNumber	14
2.8.5	hoursPerWeek	14
2.9	usesBook	14
2.9.1	CRN	14
2.9.2	semester	14
2.9.3	year	14
2.9.4	ISBN	15
2.10	prefers	15
2.10.1	rNumber	15
2.10.2	courseCode	15

2.10.3	catalogYear	15
2.10.4	rating	15
2.11	requests	16
2.11.1	rNumber	16
2.11.2	courseCode	16
2.11.3	catalogYear	16
2.11.4	justification	16
2.12	Accounts	16
2.12.1	rNumber	17
2.12.2	password	17
2.12.3	permissionType	17
3	SQL Pseudocode	18
3.1	Faculty	18
3.2	Instructor	20
3.3	Business	22
4	Design and Pseudocode of the System	30
4.1	Home Page	32
4.1.1	Back End	32
4.1.2	Front End	33
4.2	General Webpage	36
4.2.1	Glossary	36
4.2.2	Typical Interactions with a Webpage	37
5	User Interface Design	39
5.1	Wireframe	39
5.1.1	Sign-In	39
5.1.2	Edit Template	40
5.1.3	View Template	42
5.2	Typography	42
5.3	Color	43
6	Contributions	44
6.1	Ben Albot	44
6.2	Diego Lazo	45
6.3	Rex Keen	45
6.4	Eric Sanchez	46

6.5	Tara Parker	46
-----	-----------------------	----

Chapter 1

UML Data Model



Chapter 2

Relation Schemas

2.1 Courses

Courses(courseCode, catalogYear, courseTitle, required) Tuples in the Courses table contain information about a given course during a specific catalog year.

2.1.1 courseCode

PK : int Four digit numerical identifier for a class that is given at Texas Tech University. This code is used to represent all classes of this type. There may be many sections that share a course code in a given semester. (e.g. 4000)

2.1.2 catalogYear

PK : int(4) Four digit numerical identifier for a catalog year. The first two digits represent the year that the catalog begins in and the final two digits represent the year that the catalog ends in. (e.g. We represent the academic year of 2014-2015 as 1415)

2.1.3 courseTitle

: str The description of the class referred to by this tuple.

2.1.4 required

: boolean Whether or not this class is required for a degree in Computer Science (e.g. true = required, false = not required).

2.2 Sections

Sections(CRN, year, semester, sectionNumber, type, days, startTime, endTime, enrollment, capacity, room, bldg) Tuples in the Section table contain information for a unique section of a course.

2.2.1 CRN

PK : int Unique ID for the section in a given year.

2.2.2 year

PK : int (1920 < x < 3000) Year the section is being offer.

2.2.3 semester

PK : enum (Fall/Spring/Summer I/Summer II) Semester the section is being offer.

2.2.4 sectionNumber

: str The unique number of the section.

2.2.5 type

:str Teaching method of the section (e.g. online, in-class and lab).

2.2.6 days

: str Days the section meets. (e.g. MWF means the class will meet Monday, Wednesday and Friday)

2.2.7 startTime

: int ($0000 \leq x < 2400$) Time the section starts, the use of military time is for data storage efficiency and accuracy. Twelve hour time will be display for the user.

2.2.8 endTime

: int ($0000 \leq x < 2400$) Time the section ends, the use of military time is for data storage efficiency and accuracy. 12 hour time will be display for the user.

2.2.9 enrollment

: int Number of students enroll in the section.

2.2.10 capacity

: int Maximum number of students can enroll in the section according to the department.

2.2.11 room

: str The room number where the class is going to be located.

2.2.12 bldg

: str The building where the class is going to be located.

2.2.13 descriptions

: str The title of a special topics or special project course.

2.3 Instructors

Instructors(rNumber, lastName, firstName, instructorTitle, tenured, joiningSemester, joiningYear, loadPreference) Tuples in the Instructors table contain information for instructors.

2.3.1 rNumber

PK: int Unique numerical identifier for students, staff, and professors at Texas Tech University.

2.3.2 lastName

: str The last name of the instructor.

2.3.3 firstName

: str The first name of the instructor.

2.3.4 instructorTitle

: str (Professor/FTI/GPTI/Null) Title of the instructor.

2.3.5 tenured

: boolean Represents if the instructor is tenured or not.

2.3.6 joiningSemester

: str (Fall/Spring/Summer I/Summer II) The semester in which the instructor joined the college.

2.3.7 joiningYear

: int (1920 < x < 3000) The year in which the instructor joined the college. Must be greater than 0 and less than 3000.

2.3.8 loadPreference

: str (Spring/Fall/NULL) The semester in which the instructor prefers to teach more. NULL for no preference.

2.4 TAs

TAs(rNumber, lastName, firstName) Tuples in the TAs table represent information about a particular teaching assistant.

2.4.1 rNumber

PK : int Unique numerical identifier for students, staff, and professors at Texas Tech University.

2.4.2 lastName

: str The last name of the teacher assistant.

2.4.3 firstName

: str The first name of the teacher assistant.

2.5 Textbooks

Textbooks(ISBN, bookTitle, author, publisher, edition) Tuples in the Textbooks table contain information for books used in a Course

2.5.1 ISBN

PK : big int Unique numerical identifier for textbooks.

2.5.2 bookTitle

: str Title of the textbook

2.5.3 author

: str Name of the textbook author

2.5.4 publisher

: str Publisher of the textbook

2.5.5 edition

: str Edition of the Textbook

2.6 consistsOf

consistsOf(CRN, semester, year, courseCode, catalogYear) Tuples in the consistsOf table contain information for relation between a course and a section.

2.6.1 CRN

PK: int Unique ID for the section in a given year.

2.6.2 semester

PK : enum (Fall/Spring/Summer I/Summer II) Semester the section is being offer.

2.6.3 year

PK : int (1920 < x < 3000) Year the section is being offer.

2.6.4 courseCode

: int Four digit numerical identifier for a class that is given at Texas Tech University. This code is used to represent all classes of this type. There may be many sections that share a course code in a given semester. (e.g. 4000)

2.6.5 catalogYear

PK : int(4) Four digit numerical identifier for a catalog year. The first two digits represent the year that the catalog begins in and the final two digits represent the year that the catalog ends in. (e.g. We represent the academic year of 2014-2015 as 1415)

2.7 taughtBy

taughtBy(CRN, semester, year, rNumber) Tuples in the taughtBy table contain information for relation between instructor and section of a course.

2.7.1 CRN

PK: int Unique ID for the section in a given year.

2.7.2 semester

PK : enum (Fall/Spring/Summer I/Summer II) Semester the section is being offer.

2.7.3 year

PK : int (1920 < x < 3000) year the section is being offer.

2.7.4 rNumber

PK : int Unique numerical identifier for students, staff, and professors at Texas Tech University.

2.8 HasTA

HasTA(CRN, semester, year, rNumber, hoursPerWeek) Tuples in the HasTA table represent a TA being associated with (assisting) a particular section of a course.

2.8.1 CRN

PK : int The CRN of the course the TA is assisting (cf. Section).

2.8.2 semester

PK : enum (Fall/Spring/Summer I/Summer II) Semester the section is being offer.

2.8.3 year

PK : int The year of the course the TA is assisting (cf. Section). Since CRN and year are the keys for the Section table, they are sufficient to determine which section of the course is being referred to.

2.8.4 rNumber

PK : int Unique numerical identifier for students, staff, and professors at Texas Tech University.

2.8.5 hoursPerWeek

: real The number of hours each week the TA is available to help (e.g. 16.5).

2.9 usesBook

usesBook(CRN, semester, year, ISBN) The usesBook relation gives a description of the textbook that is associated with a given class.

2.9.1 CRN

PK : int is classified as an int due to it being a collection of only numbers associated with one particular class. The CRN will reference the CRN in the relation sections, and it is in this relation due to this one particular class associated with this unique number will then uniquely use a particular textbook.

2.9.2 semester

PK : enum (Fall/Spring/Summer I/Summer II) Semester the section is being offer.

2.9.3 year

PK : int is classified as an int due to it being a collection of only numbers associated with a given school year. The year will reference the year in the relation sections, and its a unique key that depends on the CRN. The year

is part of this relation due to it being in link with CRN since a CRN is dependent on the school year that a particular number can repeat.

2.9.4 ISBN

PK : big int Unique numerical identifier for textbooks.

2.10 prefers

prefers(rNumber, courseCode, catalogYear, rating) Tuples in the prefers table contain information for relation between a professor and a course preference.

2.10.1 rNumber

PK : int Unique numerical identifier for students, staff, and professors at Texas Tech University.

2.10.2 courseCode

PK : int Four digit numerical identifier for a class that is given at Texas Tech University. This code is used to represent all classes of this type. There may be many sections that share a course code in a given semester. (e.g. 4000)

2.10.3 catalogYear

PK : int(4) Four digit numerical identifier for a catalog year. The first two digits represent the year that the catalog begins in and the final two digits represent the year that the catalog ends in. (e.g. We represent the academic year of 2014-2015 as 1415)

2.10.4 rating

: int Numerical ranking of 1, 2, 3, or null. Ranking represents an instructors desire to teach a course. 1 is the most desirable, followed by 2, then 3. A null value represents that an instructor does not wish to teach the course.

2.11 requests

requests(rNumber, courseCode, catalogYear, justification) Tuples in the requests table contain information for relation between a professor and any special request they might have for a course.

2.11.1 rNumber

PK : int Unique numerical identifier for students, staff, and professors at Texas Tech University.

2.11.2 courseCode

PK : int Four digit numerical identifier for a class that is given at Texas Tech University. This code is used to represent all classes of this type. There may be many sections that share a course code in a given semester. (e.g. 4000)

2.11.3 catalogYear

PK : int(4) Four digit numerical identifier for a catalog year. The first two digits represent the year that the catalog begins in and the final two digits represent the year that the catalog ends in. (e.g. We represent the academic year of 2014-2015 as 1415)

2.11.4 justification

: str A variable length comment. Justification is a comment written by an instructor to state their justification for wanting (or not) to teach a specific course.

2.12 Accounts

Logins(rNumber, password, permissionType) Holds account information to control viewing and editing databases on the website.

2.12.1 rNumber

PK : int Unique numerical identifier for students, staff, and professors at Texas Tech University.

2.12.2 password

: char(64) Used to validate the account.

2.12.3 permissionType

: str(Faculty/Instructor/Business) Determines what the user can view and edit.

Chapter 3

SQL Pseudocode

‘\$’ denotes user-input or gathered values from system.

3.1 Faculty

1. A department faculty/staff can login and update the following information.
 - (a) When there is a new professor / FTI / GPTI joining the department, add the information of the new people into the system. The information including the joining date (which semester of which year, e.g., spring 2013), tenured or untenured, and title (e.g., assistant/associate/full professor). However for FTI/GPTI, we don't have title or tenure information.

```
SELECT rNumber, lastName, firstName, title, tenured, joiningSemester,  
       joiningYear  
FROM Instructors;
```

```
INSERT INTO Instructors ( rNumber, firstName, lastName,  
                          instructorTitle, tenured, joiningSemester, joiningYear,  
                          loadPreference) VALUES ( $rNumber, $firstName, $lastName,  
                          $instructorTitle, $tenured, $joiningSemester, $joiningYear,  
                          $loadPreference );
```

Where `tenured` and `title` may be null for FTI/GPTI. This query inserts information about new instructors into the Instructor table. An instructor may include/update their load preference later.

- (b) For each semester and every section of an offered course, input who is the instructor, where and when the section is taught, the capacity limit of the section and the enrollment of this class. For example, for CS4354 section 001, time (10am to 10:50am) days (MWF), room (204), building (ENGCTR), capacity (60), and enrollment (35). Note that for a section of a course, there may be more than one instructor.

```
INSERT INTO Sections
VALUES ($CRN, $year, $sectionNumber, $type, $semester, $days,
$startTime, $endTime, $enrollment, $capacity))
```

For each section of a course, a department faculty/staff can input the information for the section including CRN, year, section number, type of section, semester, days of the week the section meets, start time and end time of the section, enrollment, and capacity for the class.

- (c) For each section of a course, input the TA/Grader name, and hours the TA/Grader will assist this course.

Query to determine the courses to put in the dropdown:

```
SELECT distinct courseCode
FROM consistsOf NATURAL JOIN Courses
WHERE semester = $semester AND year = $year
ORDER BY courseCode;
```

Query to fill the table with course information:

```
SELECT *
FROM Sections NATURAL LEFT OUTER JOIN hasTA NATURAL LEFT
OUTER JOIN TAs
WHERE Sections.CRN IN (
    SELECT CRN
    FROM consistsOf
    WHERE courseCode = $course AND year=$year AND semester=%semester)
```

```
AND Sections.year=$year AND Sections.semester=$semester;
```

- (d) The course list of computer science. For each course, there is an attribute of catalog whose value is year. E.g., a course with catalog 2014 means that the course is a course in 2014 catalog.

```
SELECT courseCode , courseTitle
FROM Courses
WHERE catalogYear = ' " . $year . ( $year + 1 ) . " ' ;
```

Return a list of all course codes for a given catalog year, that is requested/entered by the user.

3.2 Instructor

2. A professor can login and do the following tasks

- (a) Update her/his preferences of the courses to teach in a given academic year (e.g., Fall 2013 to Summer II 2014). As for the user interface, all courses will be displayed with the professor's preference value (1-3) for that year (if the preference value is not there yet, use the previous year's preference values). The professor can edit the values. A preference value of NULL represents that the corresponding course is not preferred by the professor.

```
INSERT INTO preferences
VALUES ($rNumber, $courseCode, $catalogYear, $rating)
```

Store preferences for each instructor for each course and the year.

- (b) Update her/his teaching load distribution (only in fall and spring) preference: more load in fall or more load in spring or don't care

```
UPDATE Instructors
SET loadPreference = $loadPreference
WHERE rNumber = $rNumber
```

The instructor can update their load preference during the school year.

- (c) Input special request for a given year: course code, title, justification (<200 words).

```
INSERT INTO Requests
VALUES ($rNumber, $courseCode, $catalogYear, $justification)
```

- (d) For each section of a course assigned to this professor in a given semester, input the text books with the following information: Text Title, Author, Edition, ISBN #, Publisher. Here we assume the assignment of the given semester is already inside the database. If the course was taught before by this professor, the most recent text information should be displayed as the default text information.

Query to determine the courses an instructor has for the semester:

```
SELECT distinct courseCode
FROM consistsOf NATURAL JOIN Courses
WHERE semester=$semester AND year=$year
ORDER BY courseCode;
```

Query to fill the table with textbook information:

```
SELECT *
FROM usesBook NATURAL JOIN taughtBy NATURAL JOIN Textbooks
NATURAL JOIN consistsOf
WHERE courseCode=$row2[courseCode] AND rNumber=$_SESSION[rNumber]
ORDER BY catalogYear desc, semester DESC LIMIT 1;
```

- (e) See the courses assigned to them in the next semester. For each course, display its course code, time, days, room and building.

```
SELECT courseCode, courseTitle, startTime, endTime, days,
room, bldg
FROM Sections NATURAL JOIN consistsOf NATURAL JOIN Courses
WHERE year = $year AND semester = '$semester' AND CRN IN
```

```
(
    SELECT CRN
    FROM taughtBy
    WHERE rNumber = $_SESSION[rNumber] AND year = $year
);
```

3.3 Business

3. A business manager can obtain the following information:

- (a) For any given instructor and a number n (normally $n=5$), list all the courses (course code, title, semester, enrollment, building), in reverse chronicle order, that the instructor has taught in the last n years.

For a given instructor and a number n , the manager may also want to know the number of distinct courses with the times a course is repeated, average enrollment of this course, average TA hours (per week) for this course, and the ratio between the average TA hours and the average enrollment of this course. As an example, the information to show is something like: (CS4354, 5, 20, 4, 0.2), (CS5285, 1, 10, 0, 0). The first tuple means that the instructor has taught CS4354 5 times with average enrollment 20, average TA hours 4 (per week). Certainly it helps if you show the information in a table with the meaning of each column (and/or row) given explicitly. The undergraduate courses should be shown before the graduate courses. A clear separation of undergraduate courses from graduate courses is preferred. The required courses should also be made distinguishable from the elective courses.

Query all courses taught by selected instructor in the last n years:

```
SELECT courseCode, courseTitle, semester, year, enrollment,
bldg
FROM (((Sections JOIN taughtBy using (CRN, semester, year))
JOIN Instructors using (rNumber)) JOIN consistsOf using
(CRN, semester, year)) JOIN Courses using (courseCode, catalogYear))
WHERE year >= (2014 - $year) AND CONCAT(lastName, ', ',
```



```

firstName) = '$instructor'
ORDER BY year DESC, CASE semester
WHEN 'FALL' THEN 1
WHEN 'Summer II' THEN 2
WHEN 'Summer I' THEN 3
WHEN 'SPRING' THEN 4 END, semester";

```

Number of distinct courses taught in the last n years:

```

SELECT count(distinct(courseCode))
FROM (((Sections JOIN taughtBy using (CRN, semester, year))
JOIN Instructors using (rNumber)) JOIN consistsOf using
(CRN, semester, year)) JOIN Courses using (courseCode, catalogYear))
WHERE year >= (2014 - $year) AND CONCAT(lastName, ', ', ',
firstName) = '$instructor';

```

ALL DISTINCT UNDERGRADUATE / REQUIRED COURSES:

```

SELECT courseCode, count(courseCode), avg(enrollment), avg(hoursPerWeek),
(avg(hoursPerWeek))/(avg(enrollment))
FROM (((((Sections JOIN taughtBy using (CRN, semester, year))
JOIN Instructors using (rNumber)) JOIN consistsOf using
(CRN, semester, year)) JOIN Courses using (courseCode, catalogYear))
LEFT OUTER JOIN hasTA using (CRN, semester, year))
WHERE year >= (2014 - $year) AND CONCAT(lastName, ', ', ',
firstName) = '$instructor' AND required = 1 AND ( courseCode
LIKE '1%' OR courseCode LIKE '2%' OR courseCode LIKE '3%'
OR courseCode LIKE '4%')
GROUP BY courseCode
ORDER BY courseCode";

```

ALL DISTINCT UNDERGRADUATE / NON REQUIRED COURSES

```

SELECT courseCode, count(courseCode), avg(enrollment), avg(hoursPerWeek),
(avg(hoursPerWeek))/(avg(enrollment))
FROM (((((Sections JOIN taughtBy using (CRN, semester, year))
JOIN Instructors using (rNumber)) JOIN consistsOf using
(CRN, semester, year)) JOIN Courses using (courseCode, catalogYear))
LEFT OUTER JOIN hasTA using (CRN, semester, year))
WHERE year >= (2014 - $year) AND CONCAT(lastName, ', ', ',

```

```

firstName) = '$instructor' AND required = 0 AND ( courseCode
LIKE '1%' OR courseCode LIKE '2%' OR courseCode LIKE '3%'
OR courseCode LIKE '4%')
GROUP BY courseCode
ORDER BY courseCode";

```

ALL DISTINCT GRADUATE / REQUIRED COURSES

```

SELECT courseCode, count(courseCode), avg(enrollment), avg(hoursPerWeek),
(avg(hoursPerWeek))/(avg(enrollment))
FROM (((((Sections JOIN taughtBy using (CRN, semester, year))
JOIN Instructors using (rNumber)) JOIN consistsOf using
(CRN, semester, year)) JOIN Courses using (courseCode, catalogYear))
LEFT OUTER JOIN hasTA using (CRN, semester, year))
WHERE year >= (2014 - $year) AND CONCAT(lastName, ', ', ',
firstName) = '$instructor' AND required = 1 AND ( courseCode
LIKE '5%' OR courseCode LIKE '6%' OR courseCode LIKE '7%'
OR courseCode LIKE '8%')
GROUP BY courseCode
ORDER BY courseCode";

```

ALL DISTINCT GRADUATE / NON REQUIRED COURSES

```

SELECT courseCode, count(courseCode), avg(enrollment), avg(hoursPerWeek),
(avg(hoursPerWeek))/(avg(enrollment))
FROM (((((Sections JOIN taughtBy using (CRN, semester, year))
JOIN Instructors using (rNumber)) JOIN consistsOf using
(CRN, semester, year)) JOIN Courses using (courseCode, catalogYear))
LEFT OUTER JOIN hasTA using (CRN, semester, year))
WHERE year >= (2014 - $year) AND CONCAT(lastName, ', ', ',
firstName) = '$instructor' AND required = 0 AND ( courseCode
LIKE '5%' OR courseCode LIKE '6%' OR courseCode LIKE '7%'
OR courseCode LIKE '8%')
GROUP BY courseCode
ORDER BY courseCode";

```

- (b) For a given n , show a table which contains the following summary information for each professor: the ratio between the total number of TA hours and the total enrollment of all undergraduate courses

(and graduate courses respectively) this professor taught in the past n years, the number of all distinct courses and the number of new courses taught in the past n years, the total number of undergraduate courses (not just distinct ones) in n years and the total number of graduate courses in n years.

TA RATIO FOR UNDERGRADUATE COURSES:

```
SELECT CONCAT(lastName, ', ', firstName),
(sum(hoursPerWeek)/sum(enrollment))
FROM (((Sections JOIN taughtBy using (CRN, semester, year)
JOIN Instructors using (rNumber)) JOIN hasTA using (CRN,
semester, year) JOIN consistsOf using (CRN, semester, year))
JOIN Courses using (courseCode, catalogYear))
WHERE year >= (2014 - $year) AND ( courseCode LIKE '1%'
OR courseCode LIKE '2%' OR courseCode LIKE '3%' OR courseCode
LIKE '4%')
GROUP BY Instructors.lastName";
```

TA RATIO FOR GRADUATE COURSES:

```
SELECT CONCAT(lastName, ', ', firstName),
(sum(hoursPerWeek)/sum(enrollment))
FROM (((Sections JOIN taughtBy using (CRN, semester, year)
JOIN Instructors using (rNumber)) JOIN hasTA using (CRN,
semester, year) JOIN consistsOf using (CRN, semester, year))
JOIN Courses using (courseCode, catalogYear))
WHERE year >= (2014 - $year) AND ( courseCode LIKE '5%'
OR courseCode LIKE '6%' OR courseCode LIKE '7%' OR courseCode
LIKE '8%')
GROUP BY Instructors.lastName";
```

DISTINCT COURSES:

```
SELECT CONCAT(lastName, ', ', firstName), count(distinct
courseCode)
FROM (((Sections JOIN taughtBy using (CRN, semester, year))
JOIN Instructors using (rNumber)) JOIN consistsOf using
(CRN, semester, year)) JOIN Courses using (courseCode, catalogYear))
WHERE year >= (2014 - $year)
```

```
GROUP BY lastName";
```

```
NEW COURSES:
```

```
SELECT CONCAT(lastName, ', ', firstName), count(distinct
courseCode)
FROM (((Sections JOIN taughtBy using (CRN, semester, year))
JOIN Instructors AS T1 using (rNumber)) JOIN consistsOf
using (CRN, semester, year)) JOIN Courses using (courseCode,
catalogYear))
WHERE year >= (2014 - $year) AND courseCode NOT IN (
    SELECT courseCode
    FROM (((Sections JOIN taughtBy using
(CRN, semester, year))JOIN Instructors AS T2 using (rNumber))
JOIN consistsOf using (CRN, semester, year))JOIN Courses
using
(courseCode, catalogYear))
WHERE year < (2014 - $year) AND T1.rNumber = T2.rNumber)
GROUP BY rNumber
ORDER BY T1.lastName";
```

```
TOTAL UNDERGRAD COURSES TAUGHT:
```

```
SELECT CONCAT(lastName, ', ', firstName), count(courseCode)
FROM (((Sections JOIN taughtBy using (CRN, semester, year))
JOIN Instructors using (rNumber)) JOIN consistsOf using
(CRN, semester, year)) JOIN Courses using (courseCode, catalogYear))
WHERE year >= (2014 - $year) AND ( courseCode LIKE '4%'
OR courseCode LIKE '3%' OR courseCode LIKE '2%' OR courseCode
LIKE '1%')
GROUP BY Instructors.lastName";
```

```
TOTAL GRAD COURSES TAUGHT:
```

```
SELECT CONCAT(lastName, ', ', firstName), count(courseCode)
FROM (((Sections JOIN taughtBy using (CRN, semester, year))
JOIN Instructors using (rNumber)) JOIN consistsOf using
(CRN, semester, year)) JOIN Courses using (courseCode, catalogYear))
WHERE year >= (2014 - $year) AND ( courseCode LIKE '5%'
OR courseCode LIKE '6%' OR courseCode LIKE '7%' OR courseCode
```

```
LIKE '8%')
GROUP BY Instructors.lastName";
```

- (c) Given a number n , for each section of a special course (CS5331/CS5332), list its title, instructor, offered date (e.g., fall 2012), and enrollment.

```
SELECT instructorTitle, lastName, firstName, semester, catalogYear,
enrollment
FROM Courses NATURAL JOIN Sections NATURAL JOIN Instructors
WHERE courseCode = $courseCode
AND year >= $currentYear - $n;
```

- (d) For any given course and a number n , list all of its offerings in the last n years. For each offering, list its section number, instructor, enrollment and date, in reverse chronicle order

Query to get all the courses:

```
SELECT distinct courseCode
FROM Courses
ORDER BY courseCode;
```

Query to get all the course offering information:

```
SELECT catalogYear, sectionNumber, firstName, lastName,
enrollment, semester, year
FROM consistsOf NATURAL JOIN Courses NATURAL JOIN Sections
NATURAL JOIN taughtBy NATURAL JOIN Instructors
WHERE courseCode=$_POST[courseSelect] and year > $targetYear
ORDER BY catalogYear DESC, semester DESC;
```

- (e) See the preferences of all professors of any given year.

```
SELECT DISTINCT rNumber , courseCode , semester , rating
FROM Prefers NATURAL JOIN consistsOf
WHERE year = $year
ORDER BY rNumber;
```

- (f) See the text(s) used by a professor for a given course (code). If the professor has taught this course several times, list all texts and their corresponding semester that are used before in reverse chronicle order.

Query to get all the courses:

```
SELECT distinct courseCode
FROM Courses
ORDER BY courseCode;
```

Query to get the textbook information:

```
SELECT concat(firstName, ' '; lastName) as name, lastName,
firstName, ISBN, bookTitle, author, publisher, edition,
year, semester
FROM Sections NATURAL JOIN consistsOf NATURAL JOIN Instructors
NATURAL JOIN taughtBy NATURAL JOIN usesBook NATURAL JOIN
Textbooks
WHERE courseCode=$_POST[courseSelect]
ORDER BY lastName, firstName, year DESC, semester;
```

- (g) Given a number n , list all summer (I and II) courses with course code, instructor and enrollment in the last n years.

```
SELECT courseCode, CONCAT(lastName, ', ', firstName), enrollment,
Sections.semester, year
FROM (((consistsOf join Sections using (crn,year)) join
taughtBy using (CRN,year)) join Instructors using (rNumber))
WHERE year >= (2014-$year) AND year<=2014 AND (Sections.semester
= 'Summer I' OR Sections.semester = 'Summer II')
ORDER BY courseCode, year;
```

- (h) Given a number n , show the following statistics:

Query to get the enrollment information:

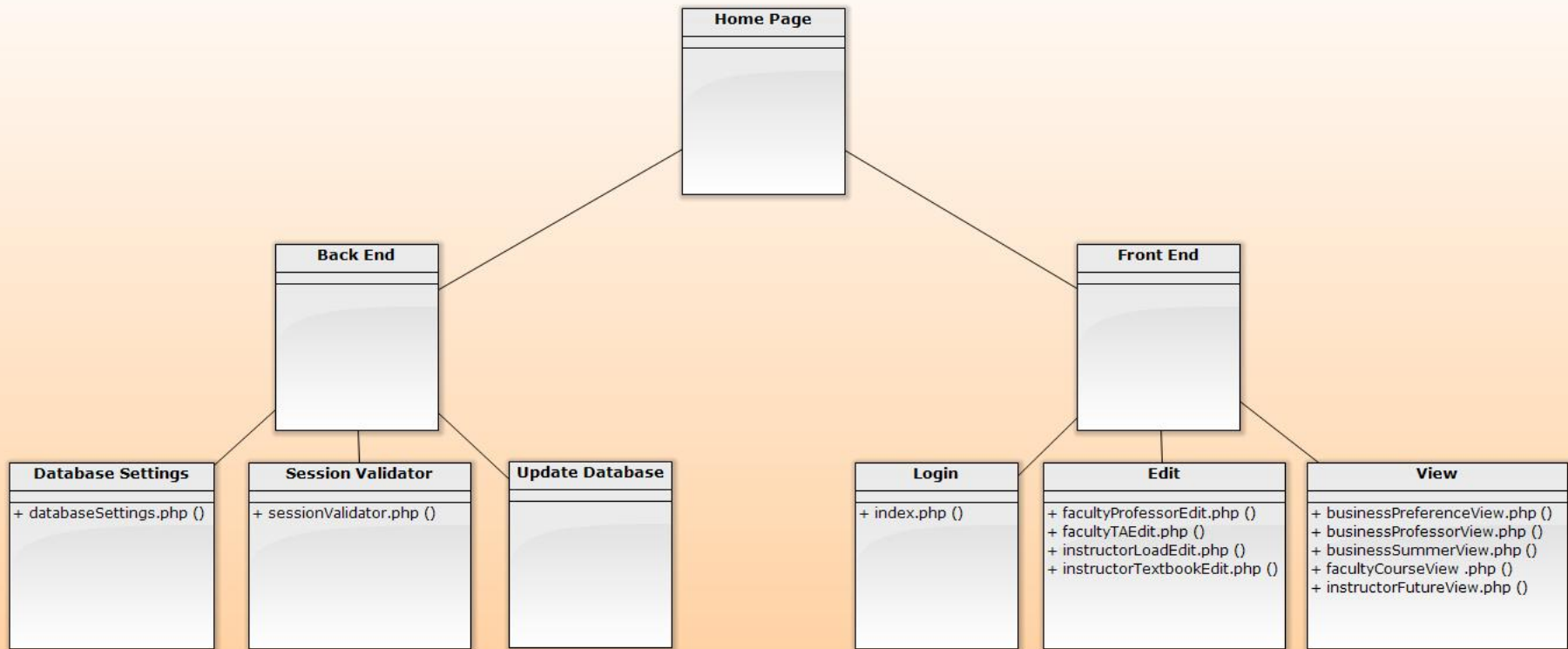
```
SELECT *
FROM
```

```
        (SELECT distinct courseCode
FROM COURSES) AS A
NATURAL LEFT OUTER JOIN
        (SELECT distinct courseCode, count(distinct CRN) AS
numClasses, sum(enrollment) AS totalEnrollment
FROM Sections NATURAL JOIN consistsOf
WHERE year > $targetYear
GROUP BY courseCode AS B;
```

Chapter 4

Design and Pseudocode of the System

The system records which instructor teaches which course(s) in a given semester with what resources, and to enable users to answer questions about the instructors and courses.



4.1 Home Page

The home page consist of a front and back end. The front end is where the application users interact with the page directly. The back end serves indirectly in support of the front end services.

4.1.1 Back End

The back end serves indirectly in support of the front end consisting in database settings, session validation and database update applications.

Database Settings

The database settings specifies the host type, username, password, and schema for the database.

Algorithm 1 Database Settings

```
1: set host type
2: set host username
3: set host password
4: set host schema
```

Session Validator

The session validator checks the rNumber of a user after their login.

Algorithm 2 Session Validator

```
1: start_session()      ▷ PHP function creates session or resumes session
2: if rNumber not valid then
3:   session_destroy()
4:   exit()
5: else
6:   if type not valid then
7:     exit()
8:   end if
9: end if
```

4.1.2 Front End

The front end is where the application users interact with the page directly consisting of login, edit and view applications.

Login

The login is the process by which user access to a page is controlled by identifying and authenticating the user's rNumber and password.

Algorithm 3 Login

```
1: start_session()
2: include database settings
3: include page layout
4: include login box
5: if rNumber and password then
6:     get int value of rNumber
7:     if rNumber > 0 then
8:         start connection between SQL and PHP
9:         if no connection error then
10:            hash password
11:            if correct password and rNumber then
12:                set expiration time for session
13:                if account type == faculty then
14:                    direct to faculty home page
15:                else if account type == instructor then
16:                    direct to instructor home page
17:                else if account type == business then
18:                    direct to business home page
19:                else
20:                    diplay error message: user has invalid account type
21:                end if
22:            else
23:                diplay error message: invalid rNumber or password
24:            end if
25:        else
26:            diplay error message: invalid request
27:        end if
28:    else
29:        diplay error message: unable to connect to database
30:    end if
31: else
32:     diplay error message: invalid rNumber
33: end if
```

Edit

The edit page is where the user has access to edit the data of the database according to the user's privileges. In the system a template is used to create all the edit pages, in addition each page is customized according to the system necessities.

Algorithm 4 Edit

```
1: set page type to (faculty|instructor)
2: include database settings
3: include session validator
4: include page layout
5: open connection with database
6: if No connection error then
7:   if Data to display then
8:     create table
9:   end if
10:  while Data to fetch do
11:    display data in table
12:  end while
13: else
14:  display error message: failure to connect
15: end if
16: get user input data
17: update database
```

View

The view page is where the user has access to view the data of the database. In the system, a template is use to create all the view pages. In addition, each page is customized according to the system necessities.

Algorithm 5 View

```
1: set page type to (faculty|business|instructor)
2: include database settings
3: include session validator
4: include page layout
5: open connection with database
6: if No connection error then
7:   if Data to display then
8:     create table
9:   end if
10:  while Data to fetch do
11:    display data in table
12:  end while
13:  close connection with database
14: else
15:   error message: failure to connect
16: end if
```

4.2 General Webpage

4.2.1 Glossary

- AJAX: Asynchronous Javascript And XML – A way for a webpage to be updated dynamically without having to be reloaded.
- HTML: HyperText Markup Language – A language for describing the information contained in webpages, as well as the links between them and the visual layout/graphics.
- HTTP: HyperText Transfer Protocol – A standard that browsers use to send and receive information to and from web servers.
- Javascript – A programming language widely used within web pages, on the client side, to make them more interactive and dynamic.
- MySQL – A popular database system, as well as a language for retrieving and storing information in databases.

- **PHP:** PHP Hypertext Preprocessor – A programming language that resides on the server side to create dynamically generated HTML webpages.
- **Server** – A computer that hosts (stores) webpages, communicating with browsers and other entities to allow them to display the webpage's information.
- **Web Browser** – A program that connects to the internet and displays webpages.
- **XML:** eXtensible Markup Language – A language similar to HTML but more general, intended for the storage of any kind of data. Often used in conjunction with programming languages for web interaction (see AJAX)

4.2.2 Typical Interactions with a Webpage

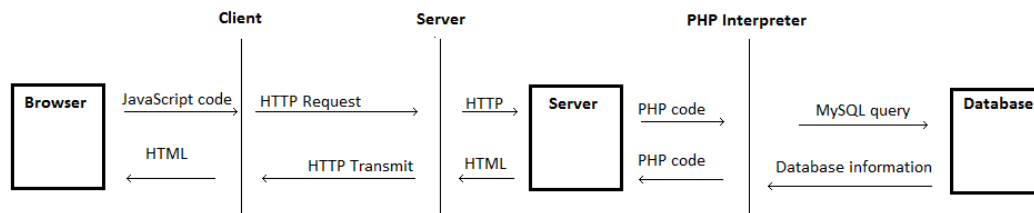


Figure 4.1: Internet Diagram

1. A browser sends an HTTP request to a server, saying it wants the information stored in the webpages.
2. The server receives the request. If the request is invalid a code will be transmitted that prompts the browser to display an error message.
3. If it is valid, the server performs different operations depending on the contents of the webpage. If the page is an HTML file, it is transmitted as is. If the page is a PHP file or contains PHP code, that code is executed on the server, generating an HTML file that is sent to the

user's browser (so the end user never sees the PHP code, only the HTML it generates).

4. If the information was correctly received, the browser displays the webpage on the screen for the user to see.
5. The user can then interact with the webpage. In our project, this mostly consists of choosing which information should be retrieved from the database and displayed – for instance, a business manager might choose which courses to display information about, or select a certain teacher from a list.
6. Once the user has made a choice, they click a button (or in some other way send a request) and some Javascript code on the page is executed.
7. If the page uses AJAX, the user's choices can be read and changes made to the webpage immediately, without sending a new request and reloading the page.
8. Otherwise, their choices are transmitted to the server, once again by HTTP request. The server then can run PHP code which uses that information as input to generate HTML code based on the user's requests, which is retransmitted to their browser.
9. Additionally, PHP code can use MySQL statements or 'queries' to retrieve information from within a database on the server, and dynamically place that information into the HTML code the user receives.

Chapter 5

User Interface Design

The section covers the basic information regarding template design and specifications. The design templates resembles the Texas Tech website template style.

5.1 Wireframe

All pages use a standardize background template. The background template contains the Texas Tech University banner on top. The background is divided into three sections, two side sections which are Tech Dark Red and one middle section which is white. The horizontal navigation bar is located at the left side just bellow the banner

5.1.1 Sign-In

The sign-in template contains the standard background template and sign-in box with no navigation bar.

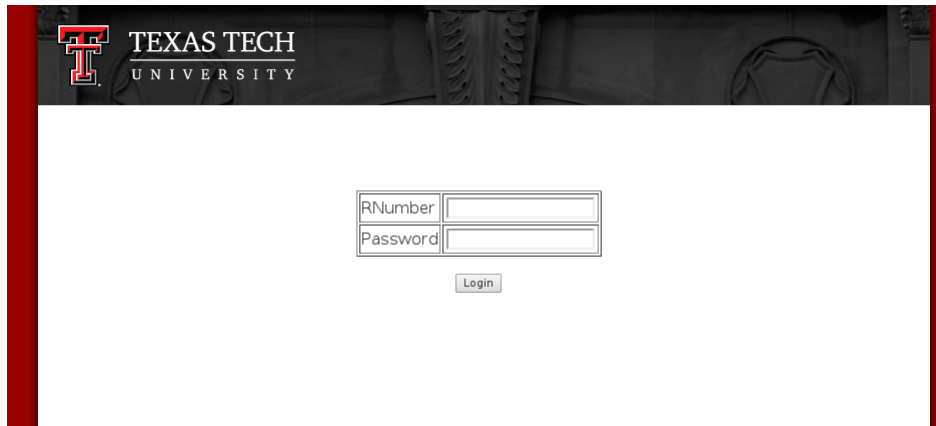



Figure 5.1: Login Page

5.1.2 Edit Template

The Edit template contains the standard background template and edit table.



TEXAS TECH

UNIVERSITY

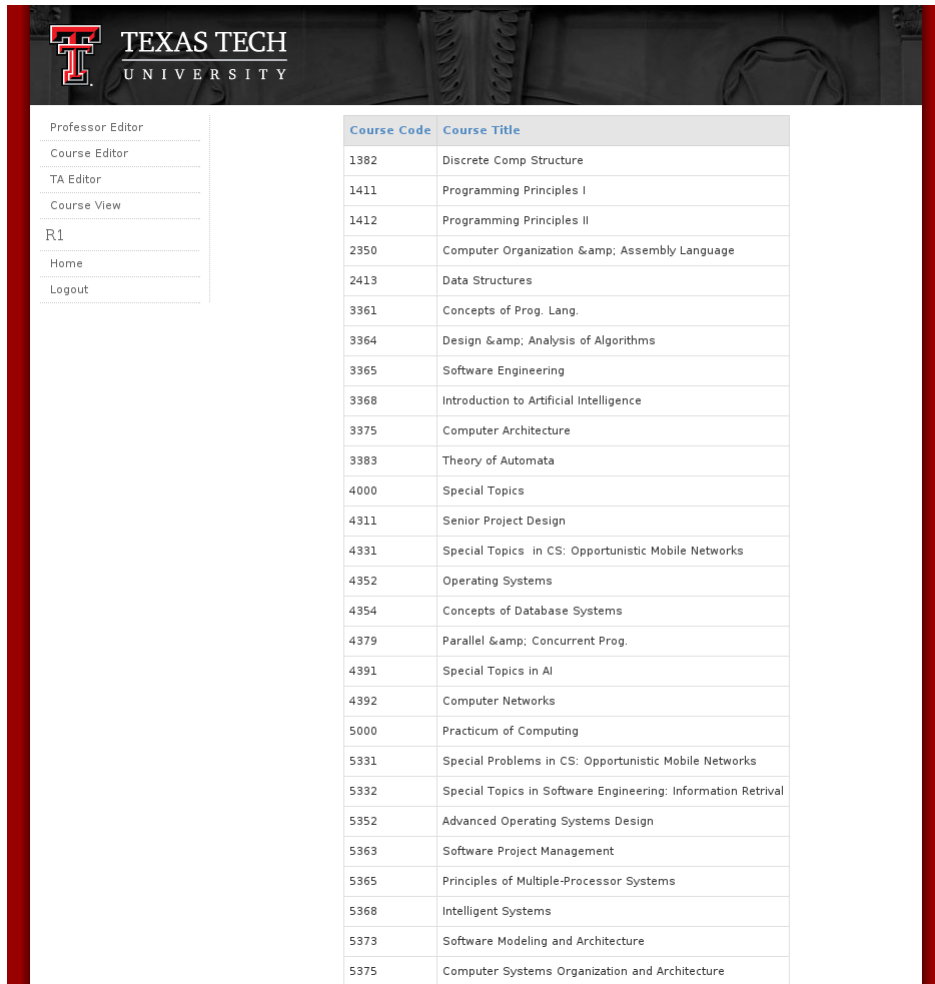
[Professor Editor](#)
[Course Editor](#)
[TA Editor](#)
[Course View](#)
[R1](#)
[Home](#)
[Logout](#)

RNumber	Last Name	First Name	Title	Tenured	Joining Semester	Joining Year
10101010	Srisinroongruang	Rattasak	Professor	<input checked="" type="checkbox"/>	Spring	2008
10226986	Rees	Josh	FTI	<input type="checkbox"/>	Spring	2011
10246785	Videtich	Amanda	Professor	<input type="checkbox"/>	Fall	2008
10326319	Zhang	Yuanlin	Professor	<input checked="" type="checkbox"/>	Fall	2009
10345678	Shin	Eonsuk	GPTI	<input type="checkbox"/>	Spring	2010
10352866	Urban	Susan		<input checked="" type="checkbox"/>	Spring	2009
10359895	Wertz	Edward	Professor	<input type="checkbox"/>	Fall	2011
10364210	Sridharan	Mohan		<input type="checkbox"/>	Fall	2009
10395487	Sobolewski	Michael	Professor	<input type="checkbox"/>	Spring	2008
10410883	Youn	Eunseog		<input checked="" type="checkbox"/>	Spring	2007
10425687	Rushton	Nelson	FTI	<input type="checkbox"/>	Fall	2009
10435487	Sinzinger	Eric	GPTI	<input type="checkbox"/>	Spring	2008
10442571	Swaminathan	Ranjini	Professor	<input type="checkbox"/>	Fall	2008
10458932	Zhuang	Yu	FTI	<input checked="" type="checkbox"/>	Summer I	2008
10465123	Pyeatt	Larry		<input type="checkbox"/>	Fall	2007
10476548	Urban	Joseph	Professor	<input type="checkbox"/>	Fall	2011
10532154	Temkin	Bharti		<input type="checkbox"/>	Spring	2010
10532457	Watson	Richard	Professor	<input type="checkbox"/>	Fall	2008
10543215	Pitalua	Mario		<input checked="" type="checkbox"/>	Spring	2009
14879548	Hewett	Rattikorn	GPTI	<input type="checkbox"/>	Spring	2007
15468752	Gelfond	Gregory	Professor	<input type="checkbox"/>	Fall	2011
16521052	Lakhani	Gopal		<input type="checkbox"/>	Fall	2007
17456874	Marcy	William	Professor	<input type="checkbox"/>	Spring	2009
18465711	Gelfond	Michael	GPTI	<input checked="" type="checkbox"/>	Fall	2011
18762034	Lamprecht	Debbie	Professor	<input type="checkbox"/>	Spring	2007
19515657	Kang	Taeghyun		<input type="checkbox"/>	Fall	2010
19624873	Lim	Sunho	GPTI	<input type="checkbox"/>	Fall	2009
19758468	Ogale	Pushkar	Professor	<input checked="" type="checkbox"/>	Spring	2010

Figure 5.2: Professor Edit Page

5.1.3 View Template

The view template contains the standard background template and output table.



Course Code	Course Title
1382	Discrete Comp Structure
1411	Programming Principles I
1412	Programming Principles II
2350	Computer Organization & Assembly Language
2413	Data Structures
3361	Concepts of Prog. Lang.
3364	Design & Analysis of Algorithms
3365	Software Engineering
3368	Introduction to Artificial Intelligence
3375	Computer Architecture
3383	Theory of Automata
4000	Special Topics
4311	Senior Project Design
4331	Special Topics in CS: Opportunistic Mobile Networks
4352	Operating Systems
4354	Concepts of Database Systems
4379	Parallel & Concurrent Prog.
4391	Special Topics in AI
4392	Computer Networks
5000	Practicum of Computing
5331	Special Problems in CS: Opportunistic Mobile Networks
5332	Special Topics in Software Engineering: Information Retrieval
5352	Advanced Operating Systems Design
5363	Software Project Management
5365	Principles of Multiple-Processor Systems
5368	Intelligent Systems
5373	Software Modeling and Architecture
5375	Computer Systems Organization and Architecture

Figure 5.3: Course View Page

5.2 Typography

The typefaces style has been written to automatically format attractive, readable headers and paragraphs with the following substitutions:

High-level headers and some major introductory paragraphs
Arial

General content and lower-level headers
Arial

5.3 Color

The page template colors are Texas Tech Dark Red and Texas Tech Black official colors. This is the primary palette used to represent Texas Tech University.

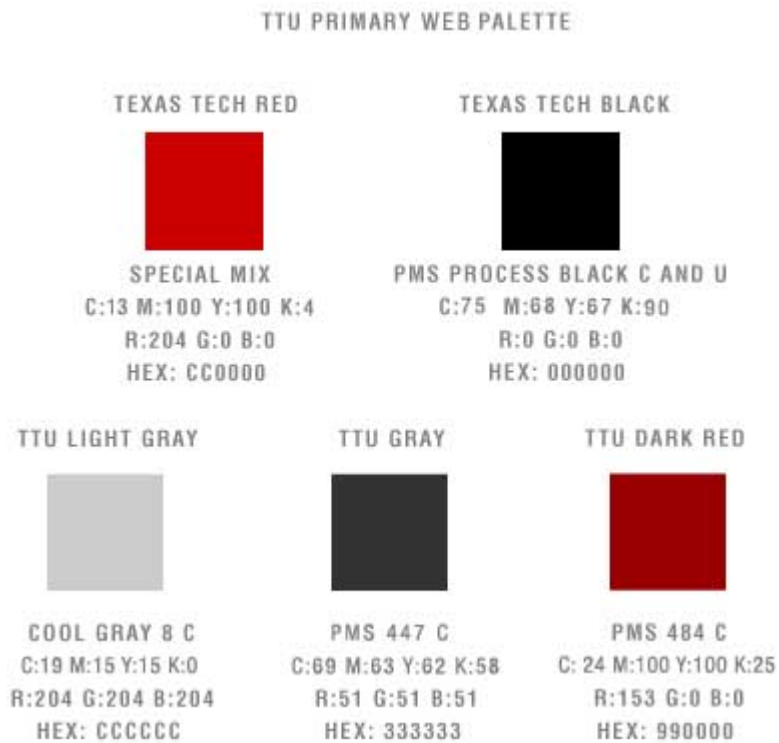


Figure 5.4: Texas Tech Color Pallet

Chapter 6

Contributions

6.1 Ben Albot

- UML diagram design/creation
- UML diagram review
- UML diagram updates
- SQL queries 2d, 2e, 3a, 3c, 3f, 3h
- Sample database
- Relation schema descriptions
- User interface mockup
- Meeting notes and blog
- Tested and modified queries 1c, 2d, 3d, 3h
- Created php pages for queries 1c, 2d, 3d, 3h
- Created diagram explaining webpage interaction for the design document

6.2 Diego Lazo

- UML diagram design/creation
- UML diagram review
- UML diagram updates
- SQL queries 1a, 1d, 2a, 2e, 3b, 3d, 3g
- Sample database textbook data
- Relation schema descriptions
- Relation schema descriptions document
- User interface document
- SQL Queries document
- Tested and modified queries 2a, 3a, 3f
- Created php pages for queries 2a, 3a, 3f
- Wrote explanations for the website diagram in the design document

6.3 Rex Keen

- UML diagram design/creation
- UML diagram review
- UML diagram updates
- SQL queries 1b, 1c, 2b, 2c, 3a, 3d, 3f
- Sample database accounts data
- Relation schema descriptions
- Research for sample data and valid input
- Prototype, professorFutureCourses.php

- Design/Pseudocode document
- Tested and modified queries 1a, 1d, 2e, 3e
- Created structure of the website
- Created template pages
- Created php pages for queries 1a, 1d, 2e, 3e
- Cleaned code and ensured proper integration from all member pages
- Implemented website design

6.4 Eric Sanchez

- SQL queries 1b, 1c, 2a, 2b, 3c, 3e, 3h
- Sample database preferences, requests, and TAs data
- Relation schema descriptions
- SQL Queries document
- Tested and modified queries 1b, 2c, 3c
- Created php pages for queries 1b, 2c, 3c

6.5 Tara Parker

- UML diagram design/creation
- UML diagram review
- UML diagram updates
- SQL queries 1a, 1d, 2c, 2d, 3b, 3e, 3g
- Relation schema descriptions
- Sample database textbooks data

- Document typesetting
- Report document and abstract
- Meeting coordinator, note taker and blog
- Tested and modified queries 2b, 3b, 3g
- Created php pages for queries 2b, 3b, 3g
- Created website design diagram (UML) for the design document