

Joel Lesko, Logan Slater, Brayden Hurl, Sri Han
CS 372
28 March, 2019
Final Project Report

UR Advisor - University of Regina Academic Advising

Problem Definition

Our project is a university class registration, advising, and grade-viewing application that takes minimal input from students, and provides them with an overview of their academic standing and helps them plan their future classes. We believed the University of Regina's current sites were slightly lacking in functionality, and significantly lacking in positive user experience. Therefore, the requirements of the problem are as follows

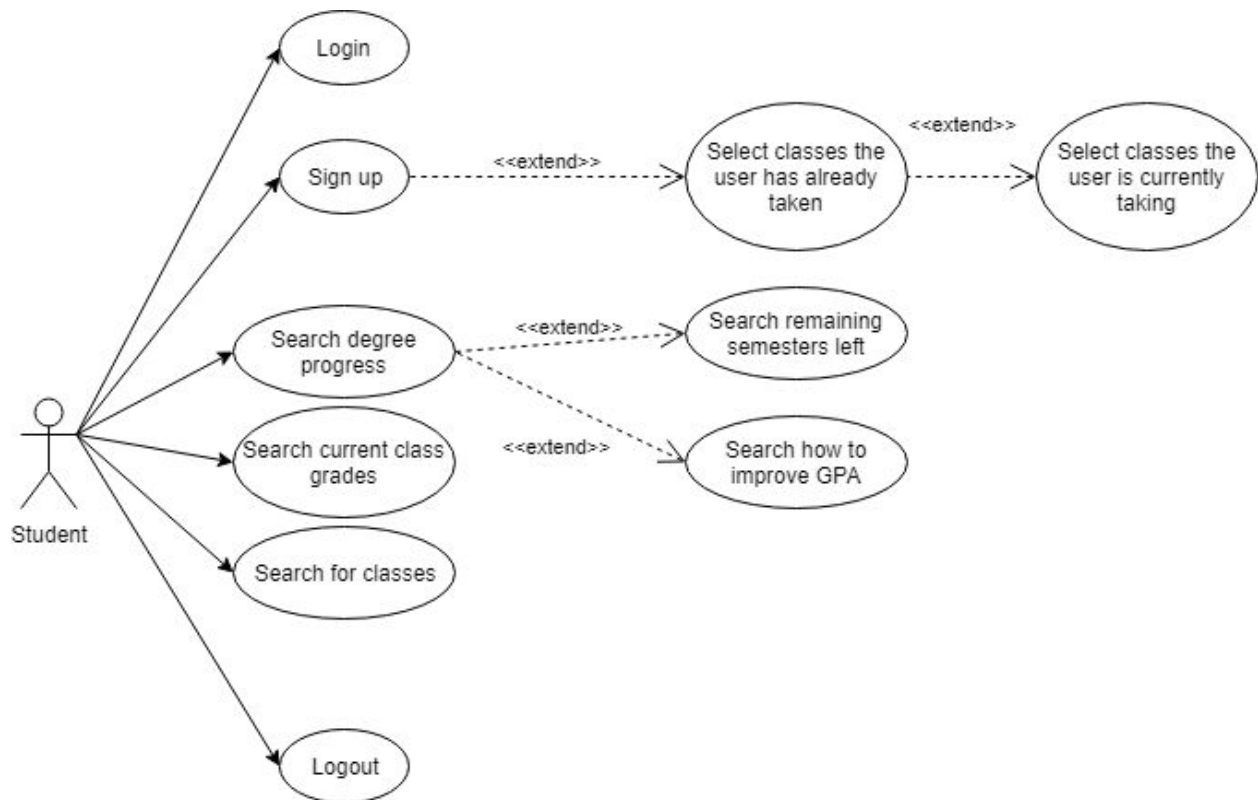
Users must be able to create new accounts, or log into existing ones. If the user is a student, they must be able to select the courses they are currently enrolled in or have already completed, view their unofficial transcript, view advice regarding their academic standing, select a degree, browse a database of available courses, register for courses, and view the marks of the assignments for their current courses. If the user is a teacher, they must be able to assign and modify grades and weights to the assignments for the course(s) they instruct.

By developing UR Advisor, we hope to host a trusted source for students in planning their overall academic schedule at a single go. Whether a student is in their first semester or their fifth, students should be able to plan their rest of the semesters easily, and receive assessments on their academic standings.

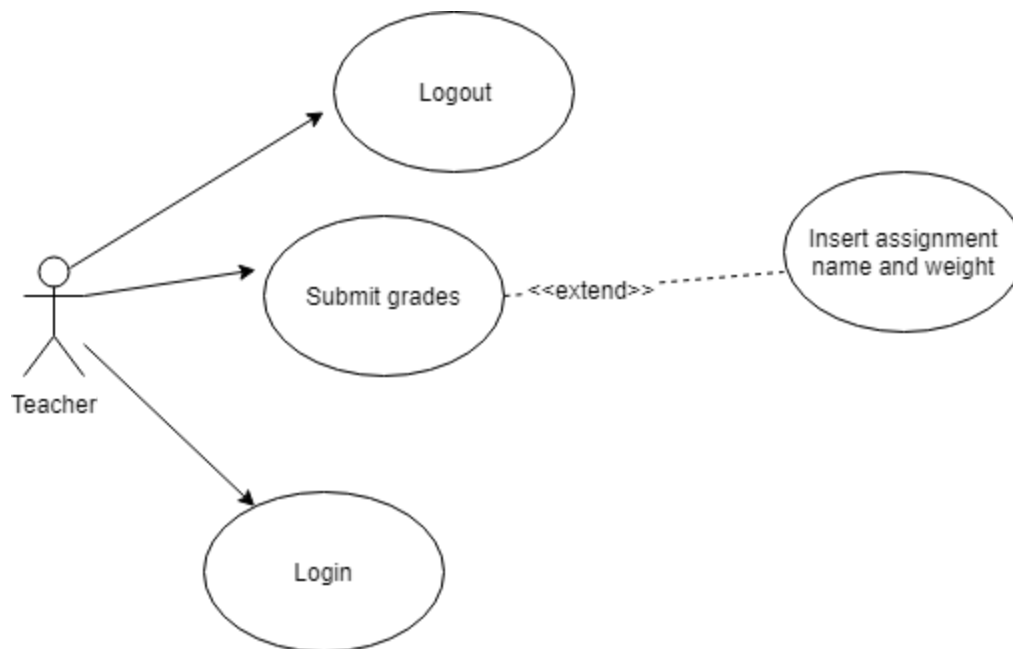
Use Cases For Users

1. Student Use Case

The student use case contains all the front-end actions the student actor can perform on the website. The student can perform basic tasks such as creating an account and logging in and out. They can perform more specific tasks such as selecting classes that they have already taken and classes that they are currently enrolled in. Also, they can use the three main features that are tailored to student users, which include searching for classes, searching for grades of their current classes, and viewing their progress towards their degree. In the the degree progression section, users can find their GPA and how many classes they have remaining.



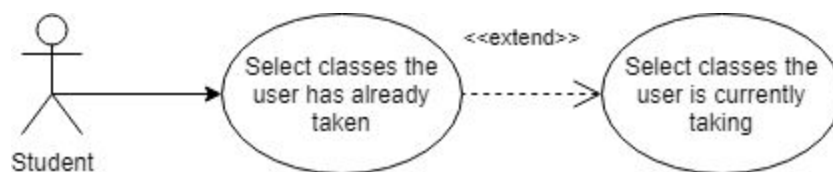
2. Teacher Use case



The Teacher Use Case diagram shows the options available to teachers on the UR advisor site. Teachers are able to perform the basics (viewing the main page, logging in and out). When signed in as a teacher they have a specific task for submitting grades for any students enrolled in the class. The site can detect the students that are enrolled automatically. All the professor has to do is insert the grade for each of the students, enter the name of the assignment and the weight. Once entered the program will check to see if the grades entered were below one hundred, if yes they will be put into the database and if not they will not be submitted to the database.

Detailed Use Cases

1. Student Selects Classes



Description: The student selects classes that they have already taken and classes that they are currently taking.

Main Actor: Student

Preconditions: The student is registered to the website.

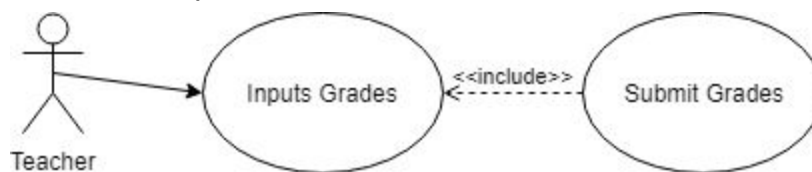
Main Flow:

1. The student selects classes that they have already completed.

- a. The student selects all classes that apply.
 - b. If the student has is a first year they can skip this step.
2. The student selects classes that they have are enrolled in.
 - a. The student selects all classes that apply.
 - b. If the student chooses to not select any classes the student will be prompted to do this again when they access the For Better Grades section.

Post conditions: The student has now selected their classes and can use the websites services.

2. Teacher Inputs Grades



Description: The teacher enters the students' grades.

Main Actor: Teacher

Preconditions: The user needs to be a teacher and needs to be instructing a class.

Main Flow:

1. The teacher accesses the grading page for that course
 - a. A list of students enrolled in the course will appear
2. The teacher inputs the grades of the students.
 - a. Grades are rejected if they are above 100% or below 0%.

Post conditions: The teacher has submitted the grades.

3. User Searches for Classes



Description: The user searches for classes that are being offered.

Main Actor: User (student or teacher)

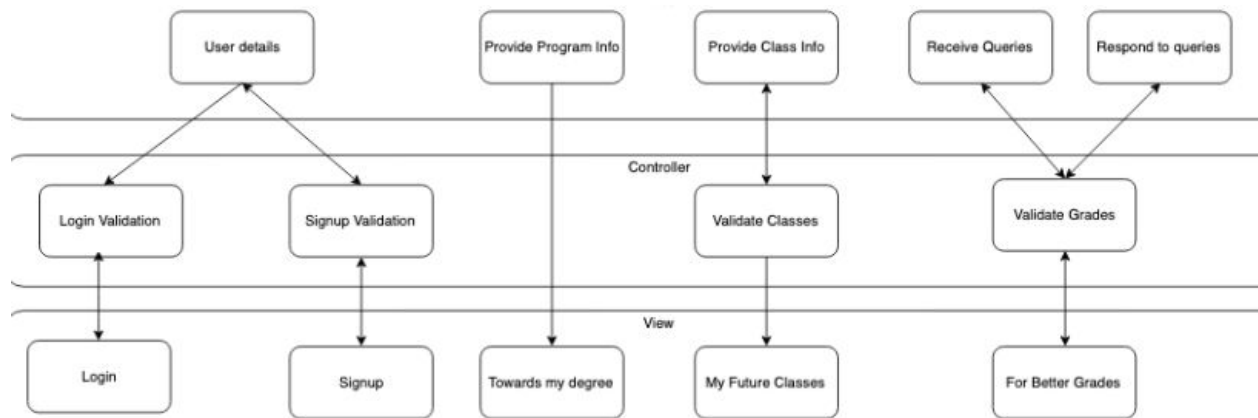
Preconditions: The user is registered to the website.

Main Flow:

1. The user accesses the search for class page.
2. The user selects the degree that they would like to see required classes for.
 - a. A list of classes is presented to the user.
 - b. The user can choose to select a class to view its information.

Post conditions: The user learnt what courses are needed for that degree.

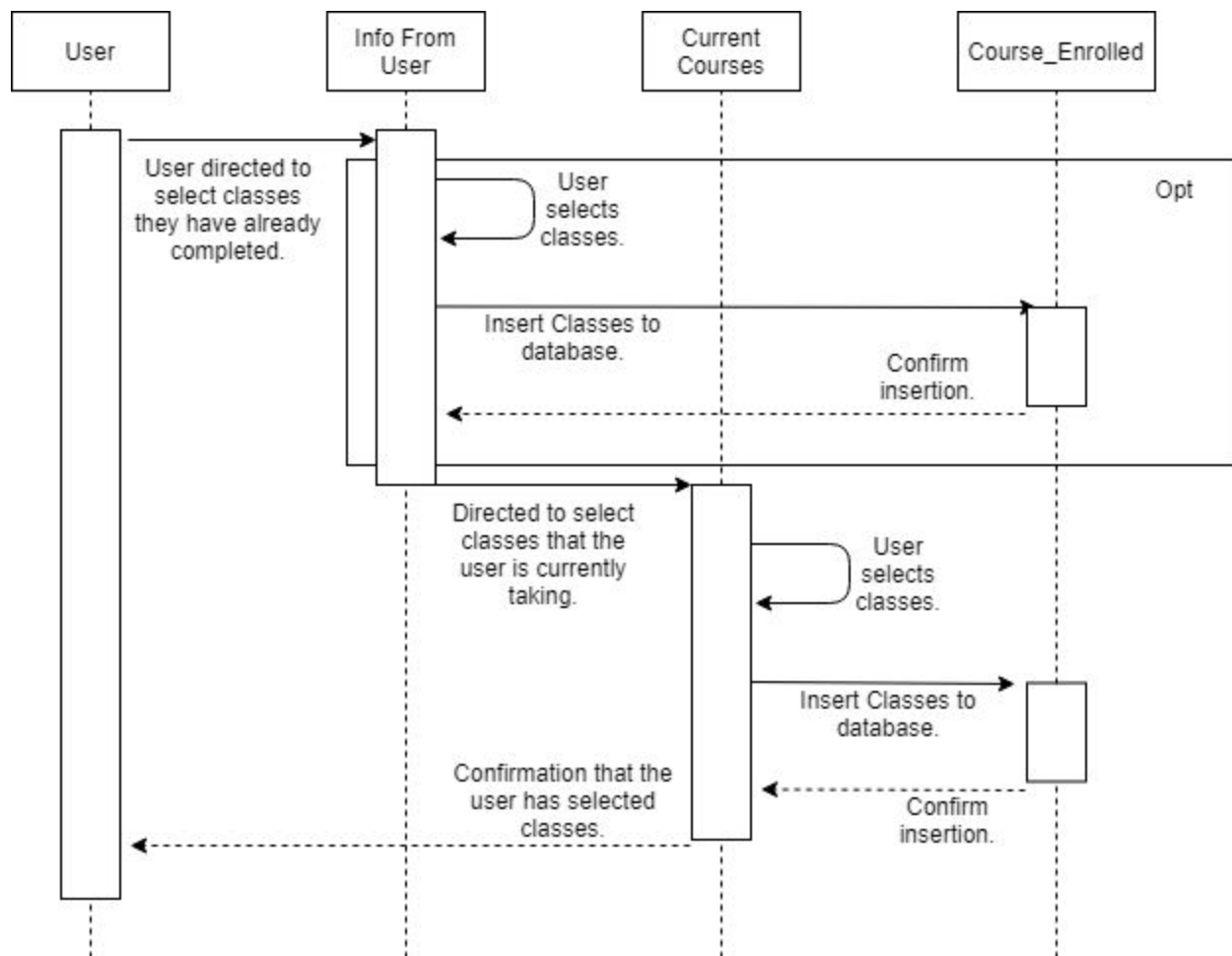
Software Architecture:



Sequence Diagrams

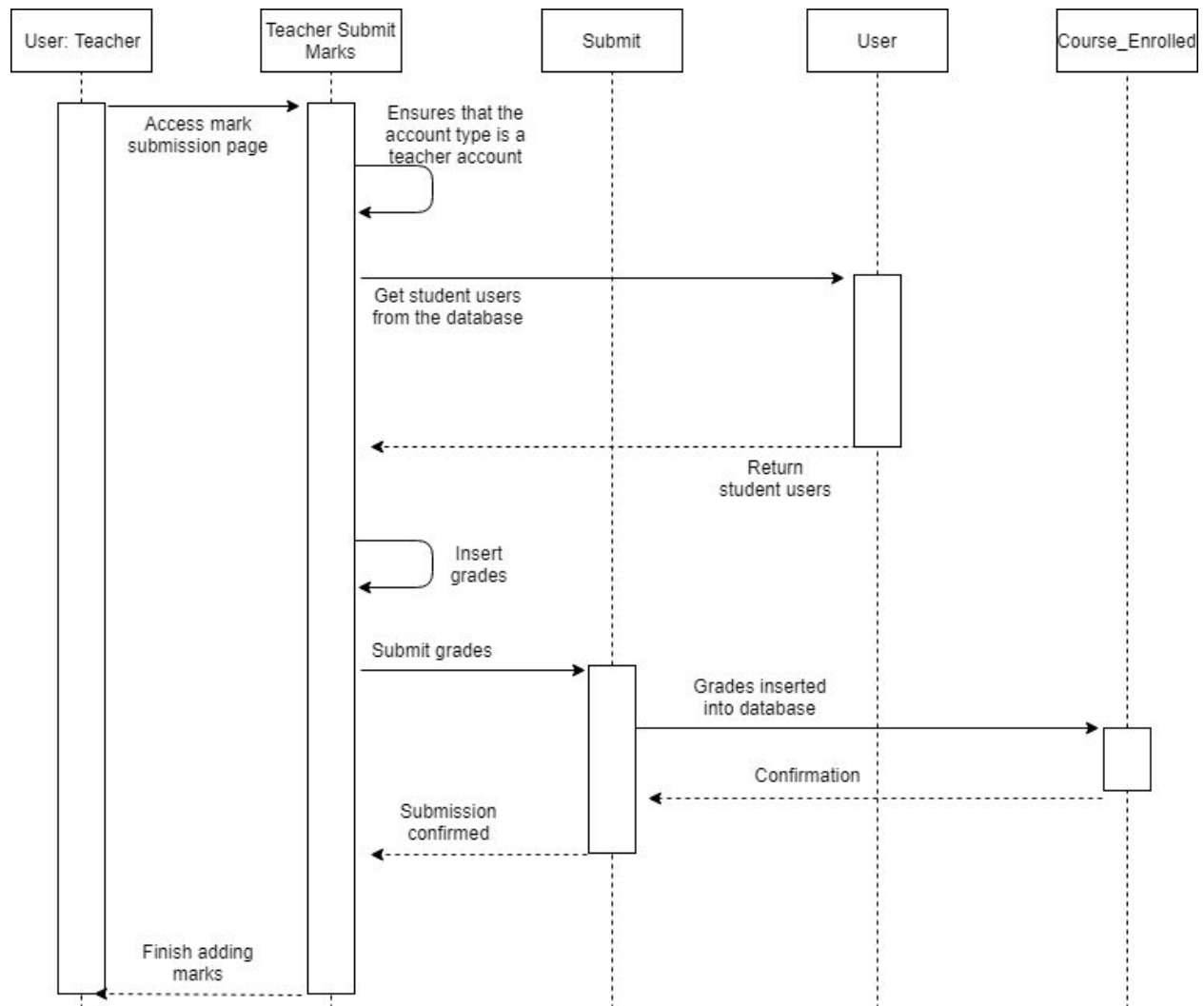
1. Student Selects Classes

The Student Selects Classes sequence begins after the user signs up to the website. The user is directed to the Info From User page, where they are presented with a list of classes, and are directed to select all classes that they have taken prior to joining the website. These classes are added to the database table `Course_Enrolled` where the classes are entered as being completed. However, this is optional. On the Current Courses page, the user is given a list of classes, and directed to select the classes they are currently enrolled in. Once the classes are selected, the database table `Course_Enrolled` is updated with them, and they are marked as currently enrolled. After this, the user can use rest of the websites services.



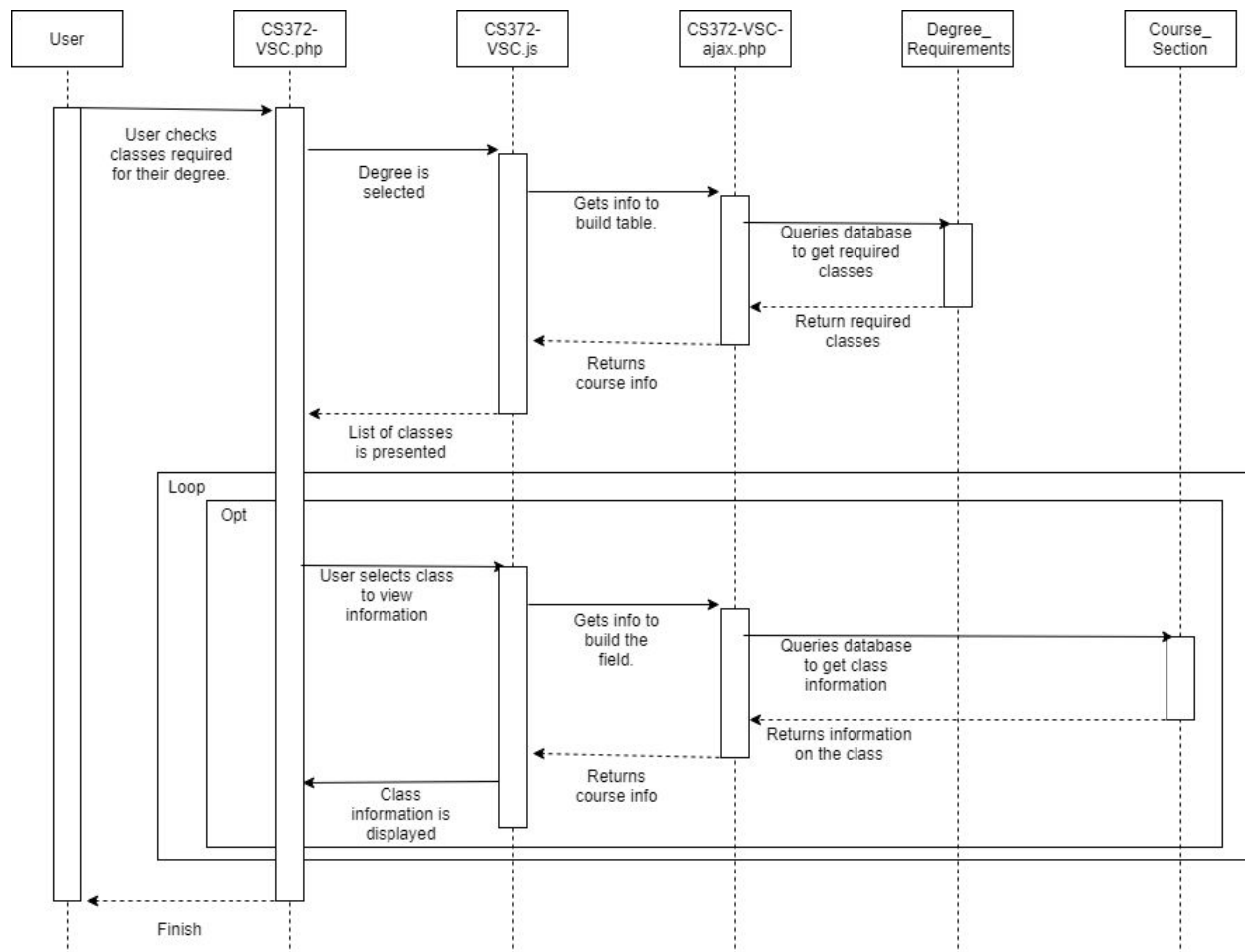
2. Teacher Inputs Grades

The teacher accesses the Teacher Submit Marks page, and the site checks that the current user is a teacher. If the user is a teacher, they are presented with student information from the database table User. Beside each student's information is a field where the teacher can enter grades. Once all grades are entered, the teacher submits the marks, which are added to entries in the Course_Enrolled table.



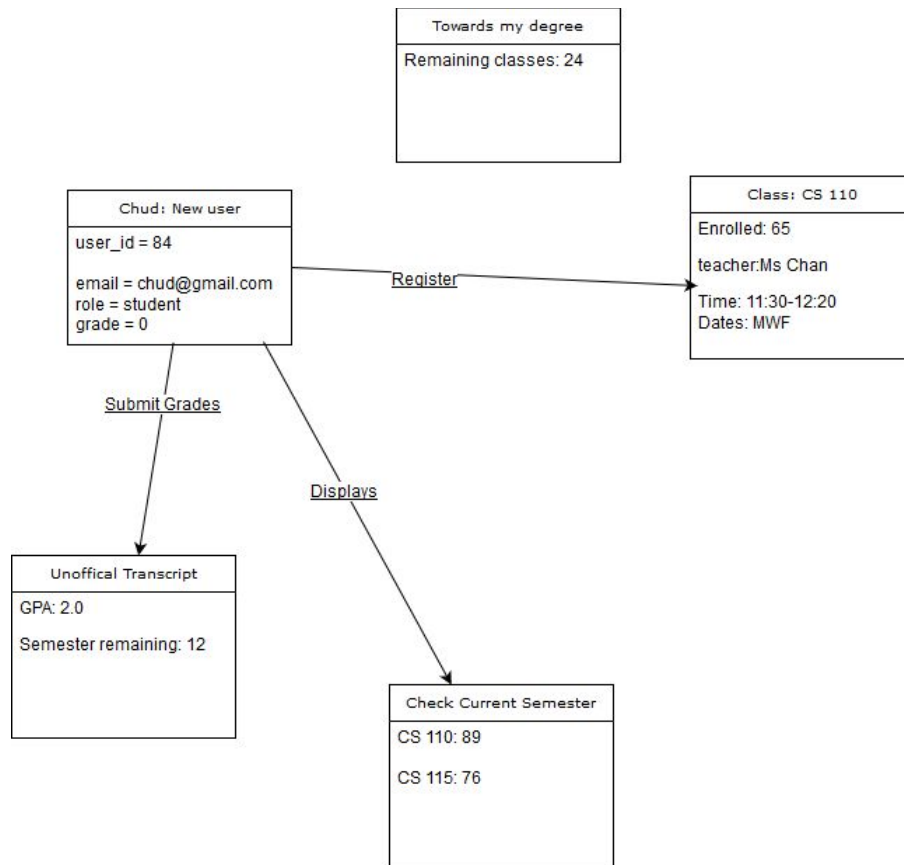
3. User Searches for Classes

The user accesses the search for classes page. Upon selecting a degree, a list is built, using AJAX to query the database, and populated with the courses required for that degree. The user can then select a course to view its detailed information, again using AJAX to query the database and add the information to the page. This process of selecting classes can be repeated any number of times until the user is done.

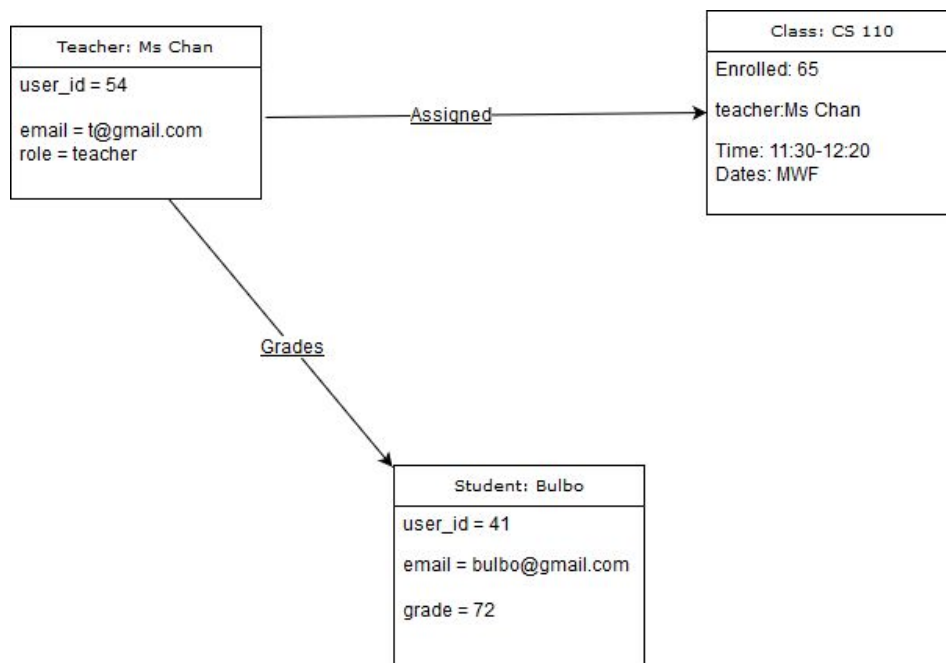


Object Diagrams

1. Student Object Diagrams



2. Teacher Object Diagram



Software Qualities

Correctness: Since we are developing an application that advises students, helps plan their future semesters, and analyzes their overall academic performance, we have tried to ensure that all information between the user and the system is accurate. Since the advice is based solely on the information provided by the user, it is important to ensure that the input data from the user is correct.

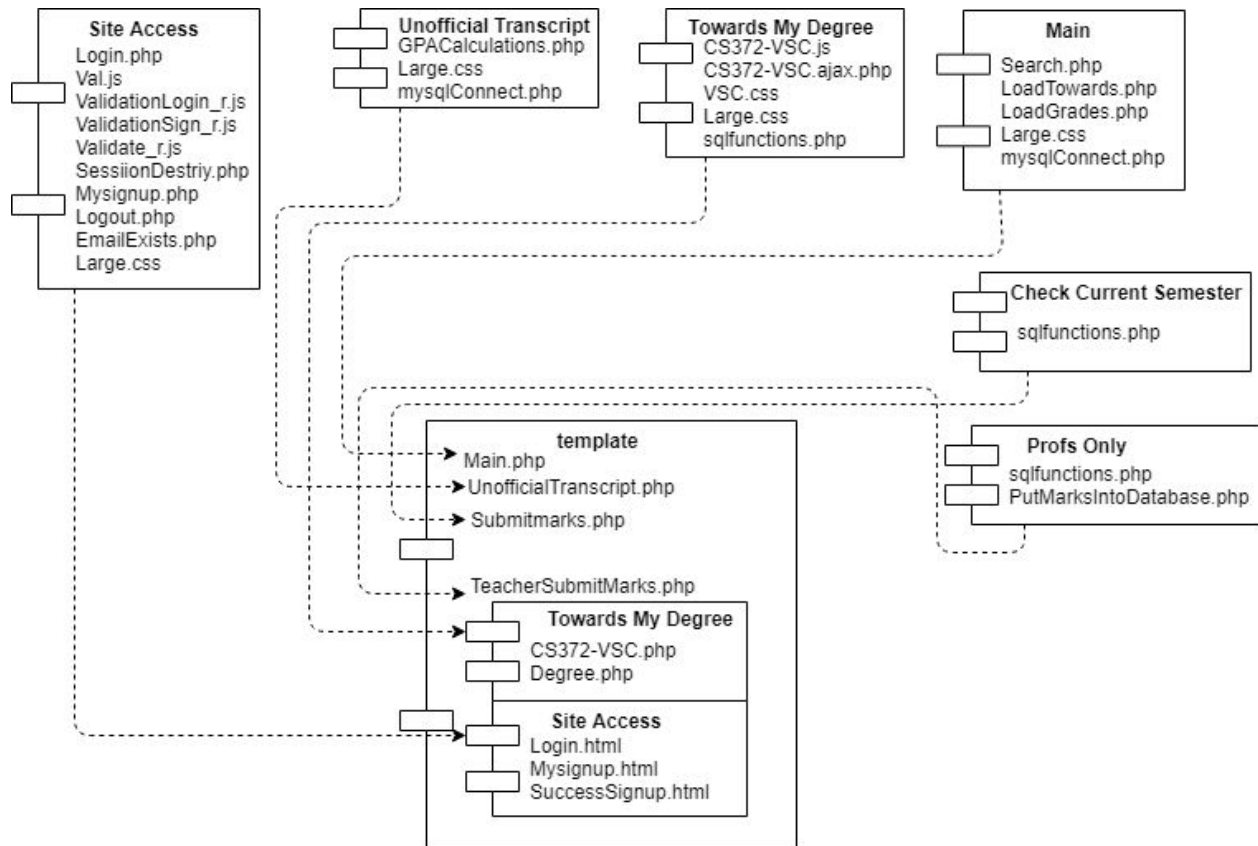
Robustness: Students are asked to enter marks for their completed courses, to generate their unofficial transcript, as well as other information. Failure to check this input from the students may cause the site to provide poor advice, and harm their academic career. To prevent this, we use JavaScript to validate all user input. We have used server-side validation for all functions that query the database. The same validation applies to instructor users as well, when they upload marks for their students.

User-Friendliness: To make it clear and easy for users to understand our application, we have tried to keep it simple: it is easy to navigate between pages, and consistent design and information layouts create a good Graphical User Interface (GUI) that provides a good user experience (UX).

Efficiency: Responsive pages are necessary for our application. Nothing on our site is particularly time-critical, but a page that loads quickly is an important part of the user's experience.

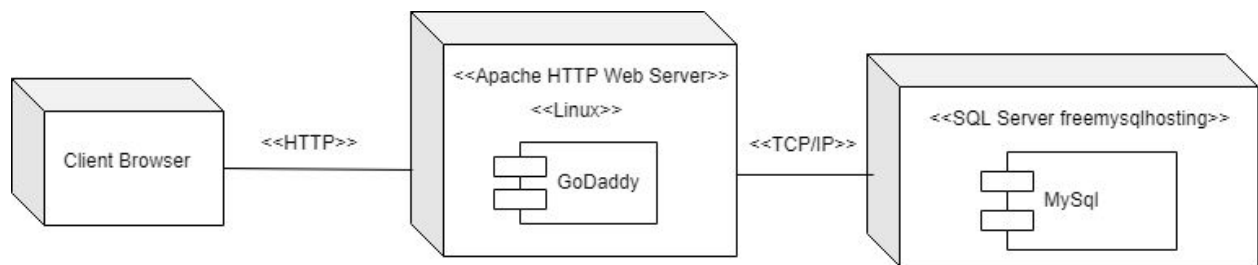
All the above mentioned software qualities are tested, and the test scenarios clearly explain in the software testing section of this document.

Component Diagram



Our website comprises of seven major components, referred to as Site Access, Unofficial Transcript, Towards My Degree, Main, Check Current Semester, Profs Only, and template. The Site Access component has all the functions and files to allow a user to access to the website. Unofficial Transcript contains the files required for a user to see their final grades, and receive feedback on how to improve their grades. Towards My Degree is a visual tool that shows and describes to the user what courses are required to complete their degree. Main is the hub of the website, where a user can navigate to other parts of the website. Check Current Semester lets the user view their grades for the classes in the current semester. Profs Only is a component of the website that can only be accessed by instructors. In Profs Only, the teacher can enter and submit the grades for their classes. The template component is the front end of all of the background processes.

Deployment Diagram



The user uses the web browser of their choice to connect to the web server, hosted on GoDaddy using HTTP. The web server queries the database using a TCP/IP connection. The database is hosted on freemysqlhosting.net SQL servers.

Implemented Functions

- CS372-VSC.js
 - function getDBterm ()
 - function checkTimeConflict (days1, start1, end1, days2, start2, end2)
 - function checkExamConflict (date1, time1, date2, time2)
 - function checkConflicts ()
 - function getTimes (start, end)
 - function getPrograms ()
 - function getCourses (newprogram)
 - function getElectives (electiveid)
 - function expandCourse (courseid)
 - function getSections (courseid)
 - function getLabs (courseid, sectionid)
 - function selectSection (courseid, sectionid, labid)
 - function displaySchedule ()
 - function changeTerm (valuechange)
- GPACalculations.php
 - function GPA(\$s, \$grade)
 - Function BySemester(\$s, \$grade, \$semSize, \$takenSemesters, \$semester)
 - function Determine(\$mark)
 - function Credits(\$s, \$grade)
 - function IncreaseGPA(\$amount, \$gpa, \$s, \$creditCount, \$modifier)
 - function Same(\$takenSemesters, \$semSize, \$semester)
 - function Remaining(\$classCount, \$amount)
- sqlfunctions.php
 - function getData(\$sql)

- function updData(\$sql)
- function sqlclose()

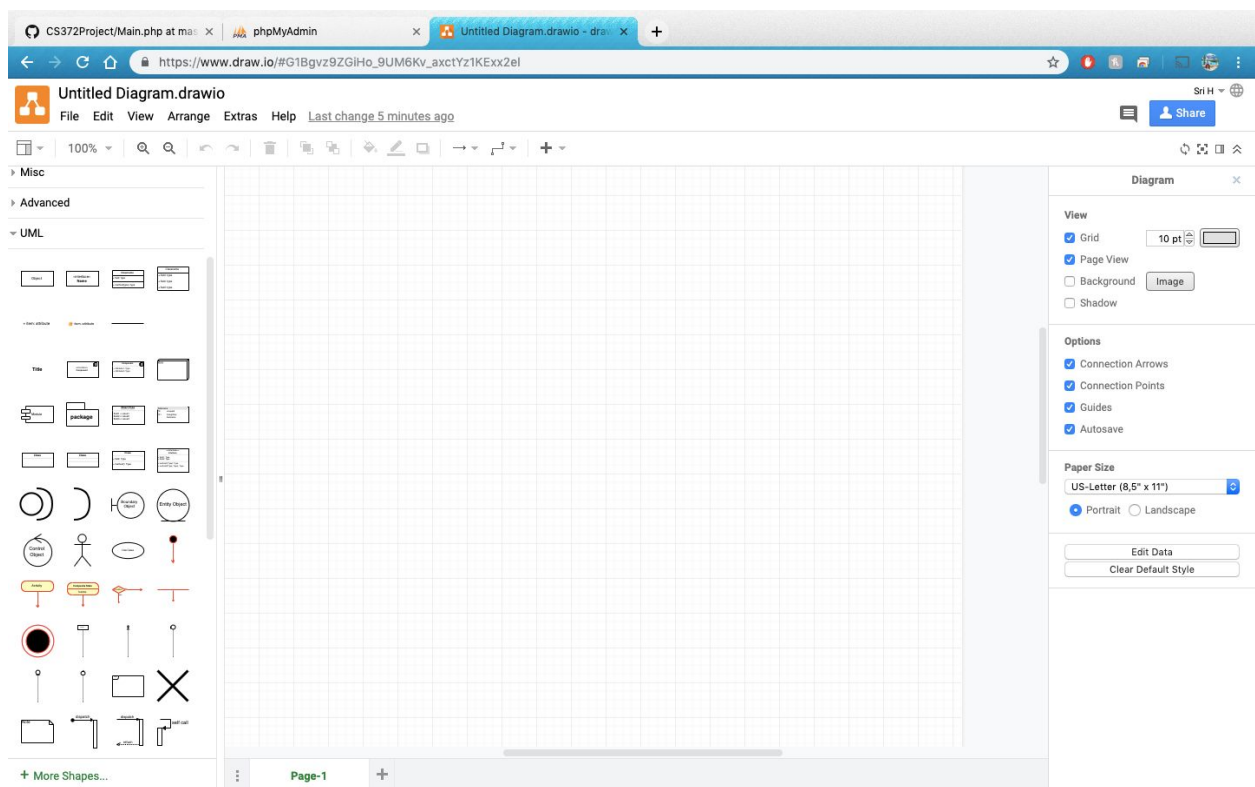
Website Link

<http://www.cs372proj.com>

Technical documentation

UML Supporting tools

For the project, we have used www.draw.io to create the use case diagrams and sequence diagrams. Draw.io is a web-based diagram editor that enables users to create tables, flowcharts, UML diagrams, and entity relation diagrams.



Programming languages

For our application UR Advisor, we have used the following languages to develop the 3 tiers:

Front-end: HTML, CSS, JavaScript

Server-side: PHP

Database: Mysql

Reused algorithms and programs

function getData(\$sql): When sending the SQL query, it stores all the result rows in an array. If there are no rows present then it returns NULL. This page is included in most of the php files that query the database. By using this function, re-writing the same code for the SQL connection for every SELECT query is unnecessary.

function updData(\$sql): This function is for INSERT and UPDATE database operations. Again, it reduces the need to rewrite the SQL statements and queries.

function sqlclose(): This function is called to close the database connection.

SessionDestroy: This is called with the onclick event of the logout button on any page.

Login: This is included on all pages, and is called when a user tries access a page without prior login.

Software tools and environments: We used XAMPP in the initial stages of our development process. We hosted our database on freemysqlhosting.net. Once we had sufficient code completed we hosted our website with GoDaddy.

Database management system

User Table

3/27/2019

www.phpmyadmin.co / sql9.freemysqlhosting.net / sql9284835 / User | phpMyAdmin 4.7.1

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	user_id	int(11)			No	None		AUTO_INCREMENT
2	degree	varchar(30)	latin1_swedish_ci		No	None		
3	student_id	int(11)			No	None		
4	school	varchar(30)	latin1_swedish_ci		No	None		
5	first_name	varchar(30)	latin1_swedish_ci		No	None		
6	last_name	varchar(30)	latin1_swedish_ci		No	None		
7	DOB	varchar(30)	latin1_swedish_ci		No	None		
8	account_type	varchar(30)	latin1_swedish_ci		No	None		
9	gpa	double			No	None		
10	email	varchar(255)	latin1_swedish_ci		No	None		
11	password	varchar(50)	latin1_swedish_ci		No	None		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	user_id	39	A	No	

Partitions

No partitioning defined!

Information

Space usage

Data	16	KiB
Index	0	B
Total	16	KiB

Row statistics

Format	Compact
Collation	latin1_swedish_ci
Next autoindex	81
Creation	Mar 23, 2019 at 10:06 AM

The User table contains data provided by the user on the signup page.

- user_id
 - Int
 - Primary key
 - A unique, auto-incrementing number assigned to each user.
- degree
 - Varchar
 - Stores what degree the user is currently working on. Inherited from Degree_Requirements table.

- student_id
 - Int
 - Stores the student's school identification number.
- school
 - Varchar
 - Name of the school the user attends.
- first_name
 - Varchar
 - The user's first name.
- last_name
 - Varchar
 - The user's last name.
- DOB
 - Date
 - User's date of birth.
- account_type
 - Varchar
 - Either "student" or "teacher" goes in this field. Will give user different access based on their role.
- gpa
 - Real
 - Stores the GPA of the user.
- email
 - varchar
 - The user's email address is stored here.
- password
 - varchar
 - Stores the user's password for their account.

Course Table

3/27/2019

www.phpmyadmin.co / sql9.freemysqlhosting.net / sql9284835 / Course | phpMyAdmin 4.7.1

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	course_section	varchar(30)	latin1_swedish_ci		No	None		
2	course_name	varchar(60)	latin1_swedish_ci		No	None		
3	type	varchar(30)	latin1_swedish_ci		No	None		
4	description	varchar(1000)	latin1_swedish_ci		Yes	NULL		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	course_section	142	A	No	

Partitions

No partitioning defined!

Information

Space usage			Row statistics	
Data	80	KiB	Format	Compact
Index	0	B	Collation	latin1_swedish_ci
Total	80	KiB	Creation	Mar 23, 2019 at 09:51 AM

Course table filled by the admin. List of data for all courses offered by the university. It is the Admin's role to keep this data up to date for the users.

- course_section
 - varchar
 - Primary key
 - Stores what section of the course is being offered. EX) CS 372.
- course_name
 - varchar
 - Stores the name of the course. Ex) Software Engineering.
- type
 - varchar
 - This field indicates what type of course the class is.
- description
 - varchar
 - This field stores course the description.
- course_section
 - varchar
 - Primary key
 - Stores what section of the course is being offered. EX) CS 372.
- course_name
 - varchar

- Stores the name of the course. Ex) Software Engineering.
- type
 - varchar
 - This field indicates what type of course the class is.
- description
 - varchar
 - This field stores course the description.

Degree_Requirements table

3/27/2019

www.phpmyadmin.co / sql9.freemysqlhosting.net / sql9284835 / Degree_Requirements | phpMyAdmin 4.7.1

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	degree_id	int(11)			No	None		AUTO_INCREMENT
2	course_section	varchar(60)	latin1_swedish_ci		No	None		
3	course2_section	varchar(60)	latin1_swedish_ci		Yes	NULL		
4	course3_section	varchar(60)	latin1_swedish_ci		Yes	NULL		
5	degree	varchar(60)	latin1_swedish_ci		No	None		
6	required_or_elective	varchar(30)	latin1_swedish_ci		No	None		
7	type	varchar(30)	latin1_swedish_ci		Yes	NULL		
8	level_requirement_min	double			Yes	NULL		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	degree_id	40	A	No	

Partitions

No partitioning defined!

Information

Space usage			Row statistics	
Data	16	KiB	Format	Compact
Index	0	B	Collation	latin1_swedish_ci
Total	16	KiB	Next autoindex	41
			Creation	Mar 24, 2019 at 10:43 PM

This table is also maintained by the Admin. This table stores the required list of courses for the user to obtain their desired degree.

- degree_id
 - int
 - Primary Key
 - Strictly excites for being a primary key.
- course_section
 - varchar

- Inherited from the Course table.
- course_section2
 - varchar
 - Inherited from the Course table. If there is an option to take one of multiple classes the optional class is stored in this field.
- course_section3
 - varchar
 - Inherited from the Course table. If there is an option to take one of multiple classes the optional class is stored in this field.
- degree
 - varchar
 - The name of the degree.
- required_or_elective
 - varchar
 - Stores either “required” or “elective”.
- type
 - varchar
 - Stores what kind of class is needed.
- level_requirement_min
 - double
 - Stores the minimal class level required to take that course.

Course Section

3/27/2019

www.phpmyadmin.co / sql9.freemysqlhosting.net / sql9284835 / Course_Section | phpMyAdmin 4.7.1

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	course_section_id	int(11)			No	None		AUTO_INCREMENT
2	course_section	varchar(30)	latin1_swedish_ci		No	None		
3	instructor_id	int(11)			No	None		
4	class_size	int(11)			No	None		
5	num_reg	int(11)			No	0		
6	day	varchar(30)	latin1_swedish_ci		No	None		
7	start_time	varchar(30)	latin1_swedish_ci		No	None		
8	end_time	varchar(30)	latin1_swedish_ci		No	None		
9	final_date	varchar(30)	latin1_swedish_ci		No	None		
10	final_time	varchar(30)	latin1_swedish_ci		No	None		
11	semester_offered_in	varchar(30)	latin1_swedish_ci		No	None		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	course_section_id	76	A	No	

Partitions

No partitioning defined!

Information

Space usage

Data	16	KiB
Index	0	B
Total	16	KiB

Row statistics

Format	Compact
Collation	latin1_swedish_ci
Next autoindex	77
Creation	Mar 23, 2019 at 10:10 AM

This table is maintained by admin. It stores section-specific information of what time the course is being offered, on which days, plus the final exam date and time.

- course_section_id
 - int
 - Primary Key
 - Strictly for primary key reasons.
- course_section
 - varchar
 - Stores what section of the course is being offered. EX) CS 372.
- class_size
 - int
 - Stores the max size of the class.

- num_reg
 - int
 - Stores the number of students currently registered for the class.
- day
 - varchar
 - Stores what day of the week the class is offered at.
- start_time
 - varchar
 - Stores what time the class starts at and is written in army time.
- end_time
 - varchar
 - Stores what time a class ends at and is written in army time.
- final_date
 - varchar
 - The date of the final.
- final_time
 - varchar
 - The time that the final is at and is written in army time.
- semester_offered_in
 - varchar
 - Indicates what semester the class was offered in. Ex) winter 2019 = w2019, summer 2019 = su2019, spring 2019 = sp2019, fall 2019 = f2019.

Course_Enrolled Table

3/27/2019

www.phpmyadmin.co / sql9.freemysqlhosting.net / sql9284835 / Course_Enrolled | phpMyAdmin 4.7.1

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	course_id	int(11)			No	None		AUTO_INCREMENT
2	user_id	int(11)			No	None		
3	course_section	varchar(30)	latin1_swedish_ci		No	None		
4	lab_section	varchar(30)	latin1_swedish_ci		Yes	NULL		
5	semester_offered_in	varchar(30)	latin1_swedish_ci		No	None		
6	mark	double			No	None		
7	class_average	double			No	None		
8	instructor_id	int(11)			No	None		
9	currently_enrolled	tinyint(1)			No	0		
10	completed	tinyint(1)			No	0		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	course_id	30	A	No	

Partitions

No partitioning defined!

Information

Space usage

Data	16	KiB
Index	0	B
Total	16	KiB

Row statistics

Format	Compact
Collation	latin1_swedish_ci
Next autoindex	157
Creation	Mar 23, 2019 at 10:11 AM

- course_id
 - int
 - Primary key
 - A unique number assigned to each course that the user has enrolled in.
- user_id
 - int
 - Inherited from the User table
- course_section
 - varchar
 - Inherited from the Course table
- lab_section
 - int

- Inherited from the Lab_Section table.
- semester_offered_in
 - varchar
 - Inherited from the course table.
- mark
 - double
 - The user's current mark in the class.
- class_average
 - double
 - The current average grade of the class.
- instructor_id
 - int
 - The user_id of the teacher in the system.
- currently_enrolled
 - bool
 - True if the user is currently taking the class and false if the user is not currently taking the class.
- completed
 - bool
 - True if the user has a passing grade in the class and false if the user is either in the class or did not get a passing grade.

Lab Table

3/27/2019

www.phpmyadmin.co / sql9.freemysqlhosting.net / sql9284835 / Lab | phpMyAdmin 4.7.1

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	lab_id	int(11)			No	None		AUTO_INCREMENT
2	course_section	varchar(30)	latin1_swedish_ci		No	None		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	lab_id	0	A	No	
course_section	BTREE	No	No	course_section	0	A	No	

Partitions

No partitioning defined!

Information

Space usage

Data	16	KiB
Index	16	KiB
Total	32	KiB

Row statistics

Format	Compact
Collation	latin1_swedish_ci
Next autoindex	13
Creation	Mar 23, 2019 at 10:13 AM

- lab_id
 - int
 - Primary key
 - Unique number given strictly for the primary key.
- course_section
 - varchar
 - Inherited from Course table

Lab_Section Table

3/27/2019

www.phpmyadmin.co / sql9.freemysqlhosting.net / sql9284835 / Lab_Section | phpMyAdmin 4.7.1

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	section_id	int(11)			No	None		AUTO_INCREMENT
2	course_section	varchar(30)	latin1_swedish_ci		No	None		
3	lab_id	int(11)			No	None		
4	course_section_id	int(11)			No	None		
5	lab_section	varchar(30)	latin1_swedish_ci		No	None		
6	lab_size	int(11)			No	None		
7	lab_num_reg	int(11)			No	None		
8	day	varchar(30)	latin1_swedish_ci		No	None		
9	start_time	varchar(30)	latin1_swedish_ci		No	None		
10	end_time	varchar(30)	latin1_swedish_ci		No	None		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	section_id	3	A	No	

Partitions

No partitioning defined!

Information

Space usage

Data	16	KiB
Index	0	B
Total	16	KiB

Row statistics

Format	Compact
Collation	latin1_swedish_ci
Next autoindex	4
Creation	Mar 23, 2019 at 10:08 AM

- section_id
 - int
 - Primary key
 - Unique number given strictly for the primary key.
- course_section
 - varchar
 - Inherited from the Lab table which was inherited from the Course table.
- course_section_id
 - int
 - The course_section_id from the Course_Section table.
- lab_section
 - varchar
 - The code for which lab a student can enroll in.
- lab_size
 - int

- Stores the number of students that can register for the lab.
- lab_num_reg
 - int
 - Stores the number of students currently enrolled in the lab section.
- day
 - varchar
 - Stores what day the lab section is offered on.
- start_time
 - varchar
 - Stores what time the lab starts at and is stored in army time.
- end_time
 - varchar
 - Stores what time the lab ends at and is stored in army time.

Marks Table

3/27/2019

www.phpmyadmin.co / sql9.freemysqlhosting.net / sql9284835 / Marks | phpMyAdmin 4.7.1

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	mark_id	int(11)			No	None		AUTO_INCREMENT
2	course_id	int(11)			No	None		
3	user_id	int(11)			No	None		
4	assignment_name	varchar(60)	latin1_swedish_ci		No	None		
5	weight	double			No	None		
6	grade	double			No	None		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	mark_id	2	A	No	

Partitions

No partitioning defined!

Information

Space usage		Row statistics	
Data	16 KiB	Format	Compact
Index	0 B	Collation	latin1_swedish_ci
Total	16 KiB	Next autoindex	9
		Creation	Mar 23, 2019 at 10:14 AM

- mark_id
 - int
 - Primary key

- A unique number assigned to a given mark. Strictly for there to be a primary key.
- course_id
 - int
 - Inherited from the Course_Enrolled table which got it from the Course table.
- user_id
 - int
 - Inherited from the Course_Enrolled table which got it from the User table.
- assignment_name
 - varchar
 - Name of the assignment.
- weight
 - double
 - Stores how much an assignment is worth for the total grade.
- grade
 - grade
 - The grade the user got on an assignment.

Requirements Table:

3/27/2019

www.phpmyadmin.co / sql9.freemysqlhosting.net / sql9284835 / Requirements | phpMyAdmin 4.7.1

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	requirements_id	int(11)			No	None		AUTO_INCREMENT
2	course_section	varchar(30)	latin1_swedish_ci		No	None		
3	required_course1	varchar(60)	latin1_swedish_ci		No	None		
4	required_course2	varchar(60)	latin1_swedish_ci		Yes	NULL		
5	required_course3	varchar(60)	latin1_swedish_ci		Yes	NULL		
6	grade_requirement	double			Yes	NULL		
7	type	varchar(30)	latin1_swedish_ci		Yes	NULL		
8	level_requirement_min	varchar(30)	latin1_swedish_ci		Yes	NULL		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	requirements_id	174	A	No	

Partitions

No partitioning defined!

Information

Space usage

Data	16	KiB
Index	0	B
Total	16	KiB

Row statistics

Format	Compact
Collation	latin1_swedish_ci
Next autoindex	175
Creation	Mar 23, 2019 at 10:09 AM

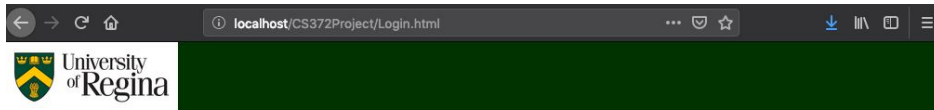
- requirement_id
 - int
 - Primary key.
 - A unique number assigned strictly for primary key reasons.
- course_section
 - varchar
 - Inherited from the Course table.
- course_name
 - varchar
 - Inherited from the Course table.
- required_course1
 - varchar
 - Course section that is needed to in order to take the class. Ex) CS 110. If no class is needed a NULL or void will be set there.
- required_course2

- varchar
- Course section that is needed to in order to take the class. This one is if there is an option between two different courses that the user can take.
Ex) CS 110. If no class is needed a NULL or void will be set there.
- required_course3
 - varchar
 - Course section that is needed to in order to take the class. This one is if there is an option between two different courses that the user can take.
Ex) CS 110. If no class is needed a NULL or void will be set there.
- grade_requirement
 - double
 - The grade you need in a previous class in order to take the class. If no grade is needed a NULL or void will be there.
- type
 - varchar
 - The type of class the user might need to take in order to that the course.
- level_requirement_min
 - varchar
 - The minimal course level required to take a class.

User Manual

User Manual: This User Manual explains what the application is, how it is useful and the different features that we implemented.

Login Page: The user either logs in to the application if they have an existing account or they will have to register to the application if they do not have access already.



Welcome to UR Private Advisor

Log In

Email

Password

Login

Do not have an account? [Sign Up](#)

Signup Page:



Sign Up Page

First Name:

Last Name:

Email:

Date Of Birth:

Password:

SignUp

[Home Page](#)

Already have an account? [Click Here](#)

Select courses you have already completed: Once the user signs up, we land on the InfoFromUser.php. The students can select the list of courses they have already completed.

localhost/CS372Project/InfoFromUser.php 50%

University of Regina

UR Main Page UR Self Serve UR Courses My Account

Select classes you have already completed:

- ☐ CS 110 - Programming and Problem Solving
- ☐ CS 115 - Object-Oriented Design
- ☐ CS 201 - Introduction to Digital Systems
- ☐ CS 210 - Data Structures and Abstractions
- ☐ CS 215 - Web Oriented Programming
- ☐ CS 280 - Risk and Reward in the Information Society
- ☐ CS 301 - Digital Systems Architecture
- ☐ CS 310 - Discrete Computational Structures
- ☐ CS 320 - Introduction to Artificial Intelligence
- ☐ CS 330 - Introduction to Operating Systems
- ☐ CS 335 - Computer Networks
- ☐ CS 340 - Advanced Data Structures and Algorithm Design
- ☐ CS 350 - Programming Language Concepts
- ☐ CS 372 - Software Engineering Methodology
- ☐ CS 476 - Software Development Project
- ☐ 400-level CS course
- ☐ 400-level CS course
- ☐ MATH 105 or 110
- ☐ MATH 111
- ☐ MATH 122
- ☐ MATH 221
- ☐ STAT 160 or 200
- ☐ MATH/STAT above 199 (not Math 261 or Stat 200)
- ☐ ENGL 100
- ☐ ENGL 110
- ☐ Arts or Fine Arts elective
- ☐ Arts or Fine Arts elective
- ☐ Arts or Fine Arts elective
- ☐ Arts or Fine Arts elective
- ☐ Arts or Fine Arts elective
- ☐ Natural Science elective
- ☐ Natural Science elective
- ☐ Science, Arts, or Fine Arts elective
- ☐ Science, Arts, or Fine Arts elective
- ☐ Science, Arts, or Fine Arts elective
- ☐ Science, Arts, or Fine Arts elective
- ☐ Science, Arts, or Fine Arts elective
- ☐ Open elective
- ☐ Open elective
- ☐ Open elective (this course cannot be from the subjects of CS, Math, Stats or ACSC)
- ☐ Open elective (this course cannot be from the subjects of CS, Math, Stats or ACSC)

Continue

Release: 8.8.3 © 2011-2018 University of Regina

Once the students selects the courses they have already completed, they are also provided an option to select the courses they are currently enrolled in.

localhost/CS372Project/CurrentCourses.php 50%

University of Regina

UR Main Page UR Self Serve UR Courses My Account

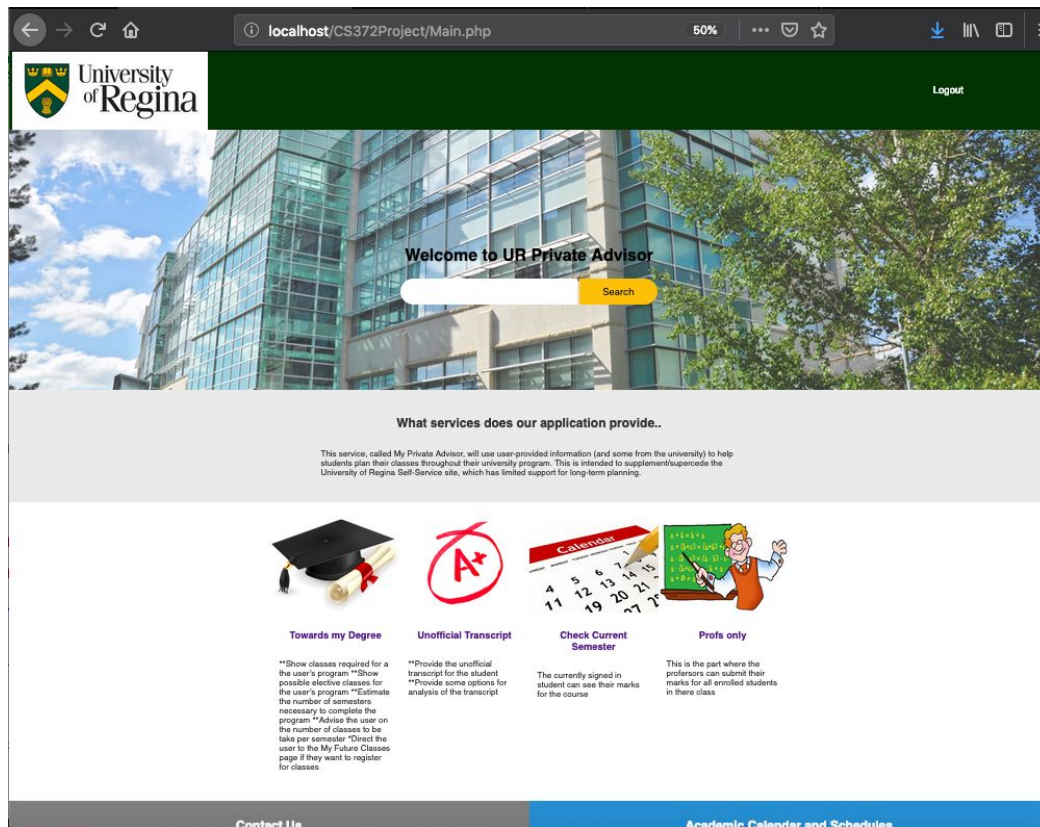
Select classes you are currently Enrolled:

- ☒ CS 110 - Programming and Problem Solving
- ☒ CS 115 - Object-Oriented Design
- ☒ CS 201 - Introduction to Digital Systems
- ☐ CS 210 - Data Structures and Abstractions
- ☐ CS 215 - Web Oriented Programming
- ☐ CS 280 - Risk and Reward in the Information Society
- ☐ CS 301 - Digital Systems Architecture
- ☐ CS 310 - Discrete Computational Structures
- ☐ CS 320 - Introduction to Artificial Intelligence
- ☐ CS 330 - Introduction to Operating Systems
- ☐ CS 335 - Computer Networks
- ☐ CS 340 - Advanced Data Structures and Algorithm Design
- ☐ CS 350 - Programming Language Concepts
- ☐ CS 372 - Software Engineering Methodology
- ☐ CS 476 - Software Development Project
- ☐ 400-level CS course
- ☐ 400-level CS course
- ☐ MATH 105 or 110
- ☐ MATH 111
- ☐ MATH 122
- ☐ MATH 221
- ☐ STAT 160 or 200
- ☐ MATH/STAT above 199 (not Math 261 or Stat 200)
- ☐ ENGL 100
- ☐ ENGL 110
- ☐ Arts or Fine Arts elective
- ☐ Arts or Fine Arts elective
- ☐ Arts or Fine Arts elective
- ☐ Arts or Fine Arts elective
- ☐ Arts or Fine Arts elective
- ☐ Natural Science elective
- ☐ Natural Science elective
- ☐ Science, Arts, or Fine Arts elective
- ☐ Science, Arts, or Fine Arts elective
- ☐ Science, Arts, or Fine Arts elective
- ☐ Science, Arts, or Fine Arts elective
- ☐ Science, Arts, or Fine Arts elective
- ☐ Open elective
- ☐ Open elective
- ☐ Open elective (this course cannot be from the subjects of CS, Math, Stats or ACSC)
- ☐ Open elective (this course cannot be from the subjects of CS, Math, Stats or ACSC)

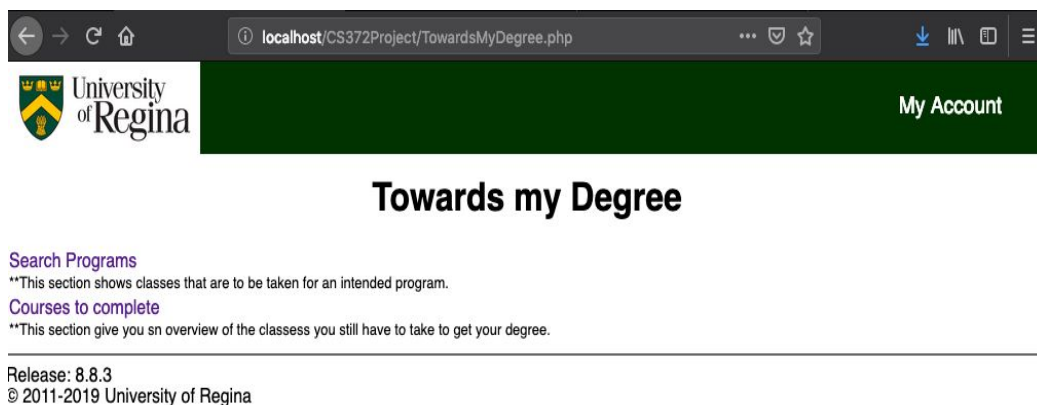
Continue

Release: 8.8.3 © 2011-2018 University of Regina

Once all the user information is taken, the student is directed to the Home page, Main.php. Students can navigate to any sections from here.

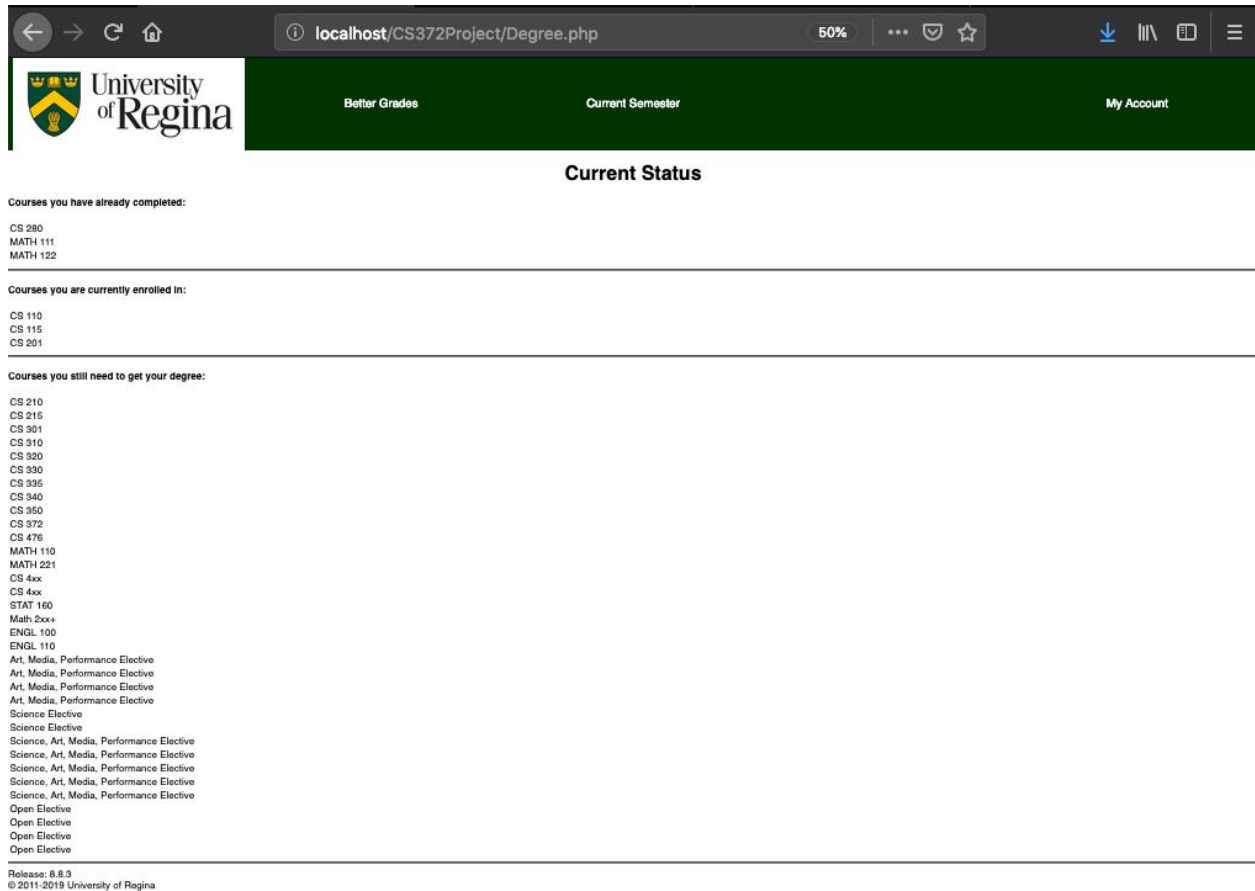


To view the complete program outline for the intended program, students can click on Towards my degree page on the HomePage and then click on Search for Program.



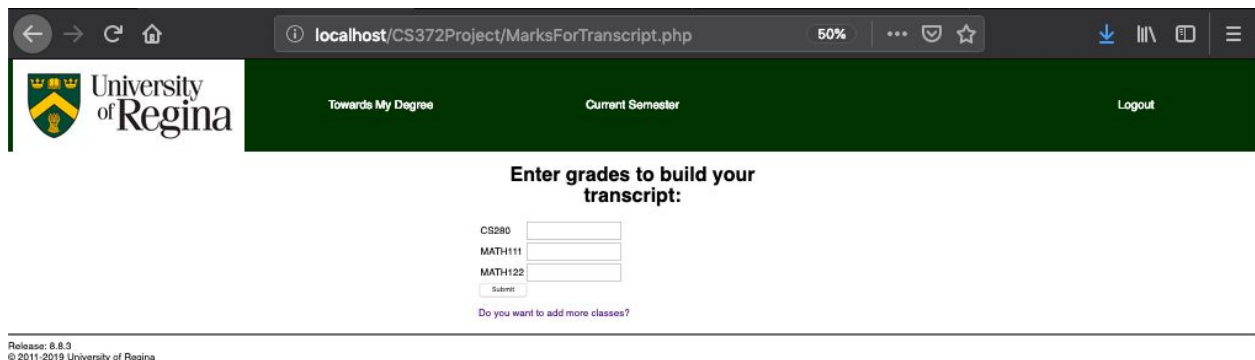
We have the unique feature for students to evaluate their overall program outline. Clicking the Courses to complete option gives a list of courses the student has already completed, list of their currently enrolled courses and the courses they are yet to complete to finish their program.

This provides a basic idea to the students on their overall course outline. This list is very accurate and efficient.




The screenshot shows a web browser window with the address bar displaying 'localhost/CS372Project/Degree.php'. The page features a green header with the University of Regina logo on the left and three navigation links: 'Better Grades', 'Current Semester', and 'My Account'. Below the header, the main content area is titled 'Current Status'. It lists three categories of courses: 'Courses you have already completed:' (CS 280, MATH 111, MATH 122), 'Courses you are currently enrolled in:' (CS 110, CS 115, CS 201), and 'Courses you still need to get your degree:' (a long list including CS 210, CS 215, CS 301, CS 310, CS 320, CS 330, CS 335, CS 340, CS 350, CS 372, CS 476, MATH 110, MATH 221, CS 4xx, CS 4xx, STAT 160, Math 2xx+, ENGL 100, ENGL 110, Art, Media, Performance Elective, Science Elective, Science, Art, Media, Performance Elective, Open Elective, etc.). At the bottom, it shows 'Release: 6.8.3' and '© 2011-2019 University of Regina'.

Our application takes inputted grades from the students and generates an unofficial transcript that gives advices to the students on their academic standing.



The screenshot shows a web browser window with the address bar displaying 'localhost/CS372Project/MarksForTranscript.php'. The page features a green header with the University of Regina logo on the left and three navigation links: 'Towards My Degree', 'Current Semester', and 'Logout'. Below the header, the main content area is titled 'Enter grades to build your transcript:'. It contains a form with three input fields for 'CS280', 'MATH111', and 'MATH122', followed by a 'Submit' button. Below the form, it asks 'Do you want to add more classes?'. At the bottom, it shows 'Release: 6.8.3' and '© 2011-2019 University of Regina'.



University

of Regina

Towards My Degree

Current Semester

Logout

Unofficial Transcript

Course	Grade
CS 280	65
MATH 111	77
MATH 122	56
Average: 66.00	


GPA Over Time

Your GPA of classes prior to our services is 2.00
 Your over all GPA is 2.00
 To increase your GPA to a 3 you need to take 3 classes and get a grade of about 80%.
 If you take 3 classes every semester it will take 13 to finish.
 If you take 4 classes every semester it will take 10 to finish.
 If you take 5 classes every semester it will take 8 to finish.

Do you want to add more classes?

Release: 8.8.3
© 2011-2019 University of Regina

Students can also check grades for their currently enrolled classes by clicking on the Check Current Semester section of the application:



University

of Regina

Towards My Degree

Transcript

Logout

My Grades

Course Name: CS 110

Assignment 1: 85/100

Assignment 2: 90/100

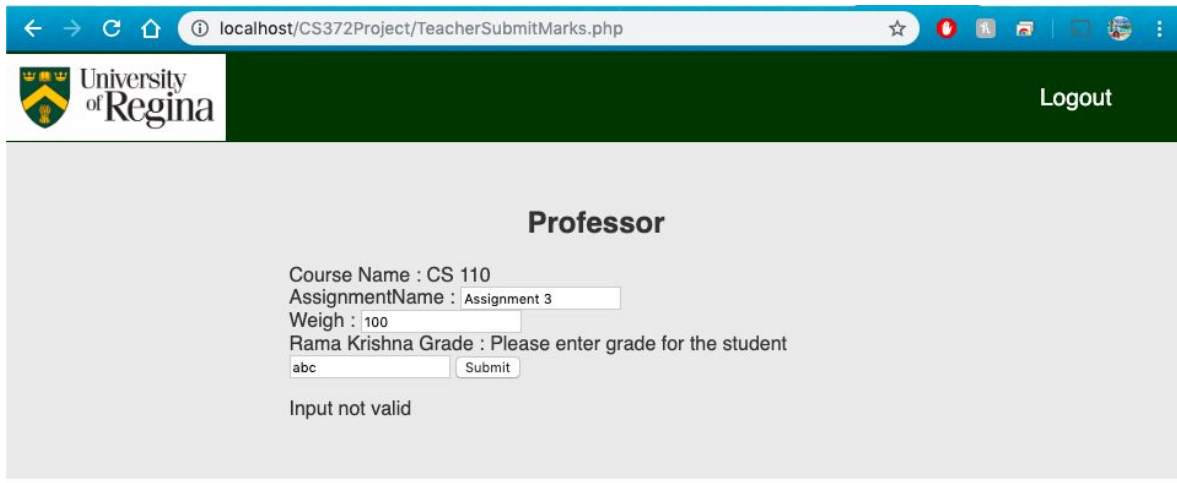
Course Name: CS 115

Assignment 1: 85/100

Assignment 2: 90/100

Release: 8.8.3
© 2011-2019 University of Regina

With the professor as the actor, we have given the ability to the professor to post grades for the students that are currently enrolled in their classes. Only the professor will have access to this section.



localhost/CS372Project/TeacherSubmitMarks.php

University of Regina

Logout

Professor

Course Name : CS 110
AssignmentName : Assignment 3
Weigh : 100
Rama Krishna Grade : Please enter grade for the student
abc Submit

Input not valid

Release: 8.8.3
© 2011-2019 University of Regina

Testing:

In this section we will concentrate on three key software qualities for this project: correctness, robustness, and performance.

Correctness:

Test Scenario:

Input action - User tries to sign up using valid credentials but the user email is already taken.

Sign Up Page

First Name:

Last Name:

Email:

Date Of Birth:

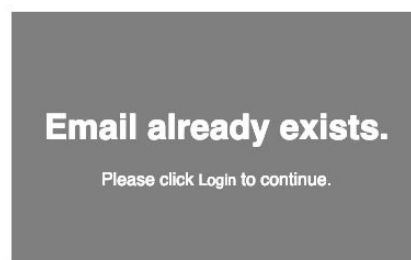
Password:

[SignUp](#)

[Home Page](#)

[Already have an account? Click Here](#)

Output Action - User get to the page, where it says the email account already exists and please go to the login screen to continue.



Correctness:

Test Scenario:

Input Action: A user tries to access the application without logging in.

Output Action: The application takes user to the login page and asks them to login or signup first.

Note: The application also recognises the type of user logged in and provides access based on their account type. If the logged in user is a student is a student, they cannot access the Prof Only section. Where as a Professor cannot access the rest of the

features other than Prof Only section. The application would redirect the users to main page, when they try to access the section they are not granted access to.

localhost/CS372Project/Main.php

What services does our application provide..

This service, called My Private Advisor, will use user-provided information (and some from the university) to help students plan their classes throughout their university program. This is intended to supplement/supercede the University of Regina Self-Service site, which has limited support for long-term planning.



Towards my Degree

**Show classes required for a the user's program **Show possible elective classes for the user's program **Estimate the number of semesters necessary to complete the program **Advise the user on the number of classes to be take per semester *Direct the user to the My Future Classes page if they want to register for classes



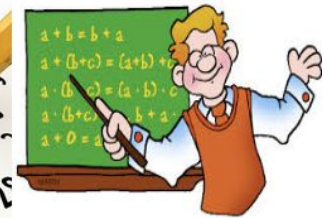
Unofficial Transcript

**Provide the unofficial transcript for the student
**Provide some options for analysis of the transcript



Check Current Semester

The currently signed in student can see their marks for the course



Profs only

This is the part where the profersors can submit their marks for all enrolled students in there class

Correctness

Test Scenario:

Input action: Once the user registers for the application and clicks on Current Semester, they will get an accurate list of remaining classes for the user to complete their program:

The screenshot shows a web browser window with the URL `localhost/CS372Project/Degree.php`. The page features a dark green header with the University of Regina logo and three navigation links: "Better Grades", "Current Semester" (which is active), and "My Account". Below the header, the main content area is titled "Current Status". It is divided into three sections:

- Courses you have already completed:**
 - CS 110
 - CS 115
 - CS 476
- Courses you are currently enrolled in:**
 - CS 201
- Courses you still need to get your degree:**
 - CS 210
 - CS 215
 - CS 280
 - CS 301
 - CS 310
 - CS 320
 - CS 330
 - CS 335
 - CS 340
 - CS 350
 - CS 372
 - MATH 110
 - MATH 111
 - MATH 122
 - MATH 221
 - CS 4xx
 - CS 4xx
 - STAT 160
 - Math 2xx+
 - ENGL 100
 - ENGL 110
 - Art, Media, Performance Elective
 - Art, Media, Performance Elective
 - Art, Media, Performance Elective
 - Art, Media, Performance Elective

Correctness:

Test Scenario:

Input action:

When the user enters their grades for their completed courses to build their unofficial transcript, the application provides an accurate response.

Enter grades to build your transcript:

CS335

CS340

Unofficial Transcript

Course	Grade
CS 110	78
CS 115	88
CS 201	65
CS 215	44
CS 280	65
CS 301	50
Average: 65.00	

GPA Over Time

Your GPA of classes prior to our services is 2.00
Your over all GPA is 2.00
To increase your GPA to a 3 you need to take 6 classes and get a grade of about 80%.
If you take 3 classes every semester it will take 12 to finish.
If you take 4 classes every semester it will take 9 to finish.
If you take 5 classes every semester it will take 7 to finish.

Do you want to add more classes?

Unofficial Transcript

Course	Grade
CS 110	78
CS 115	88
CS 201	65
CS 215	44
CS 280	65
CS 301	50
CS 335	45
CS 340	77
Average: 64.00	

GPA Over Time

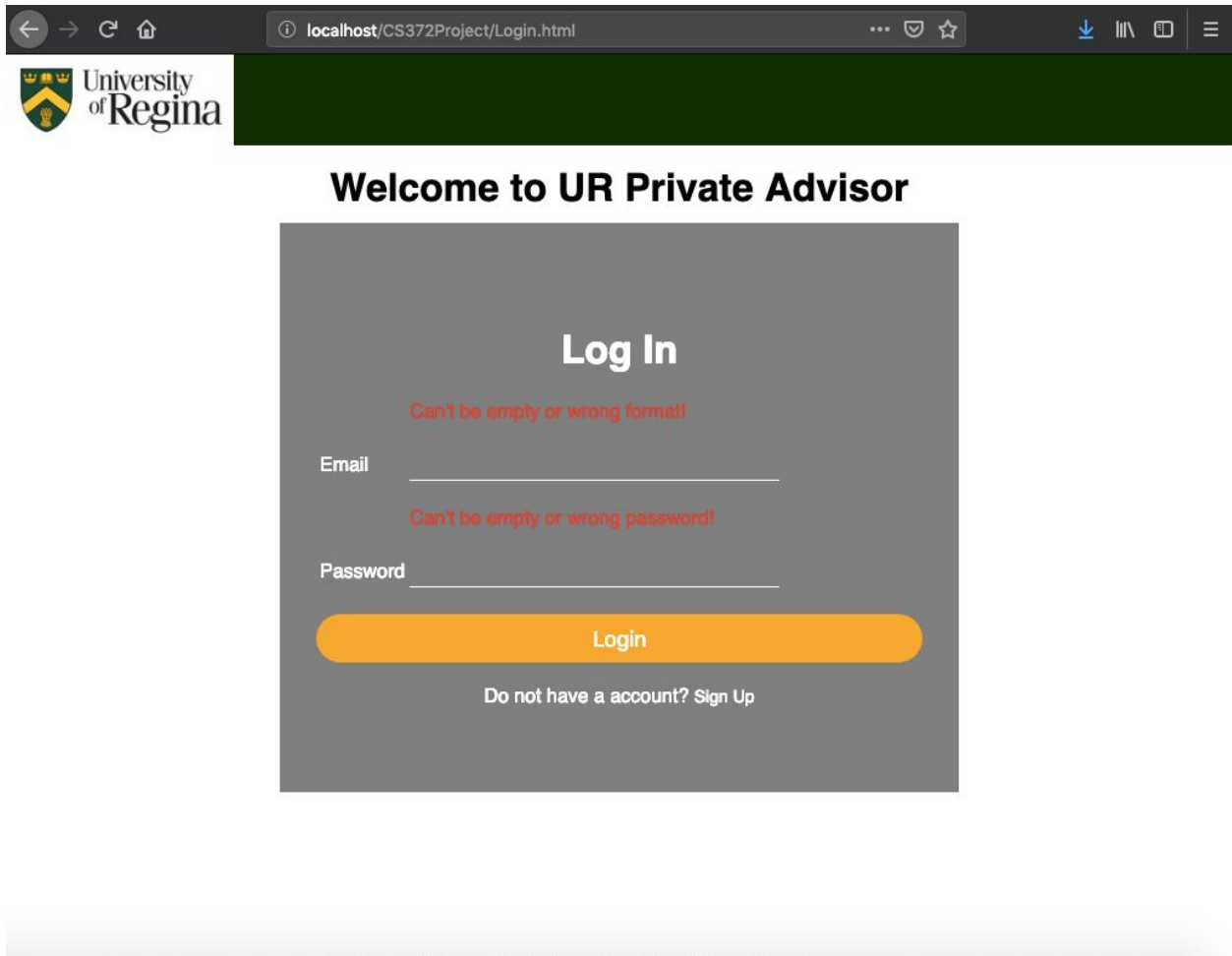
Your GPA of classes prior to our services is 1.88
Your over all GPA is 1.88
To increase your GPA to a 2 you need to take 1 classes and
get a grade of about 70%.
To increase your GPA to a 2 you need to take 1 classes and
get a grade of about 80%.
To increase your GPA to a 3 you need to take 9 classes and
get a grade of about 80%.
If you take 3 classes every semester it will take 11 to finish.
If you take 4 classes every semester it will take 8 to finish.
If you take 5 classes every semester it will take 7 to finish.

[Do you want to add more classes?](#)

Robustness:

Input action - User clicks on the login button without entering any credentials.

Output action - The system would detect the invalid input and displays the error messages.



The screenshot shows a web browser window with the address bar displaying "localhost/CS372Project/Login.html". The page header features the University of Regina logo and name. The main content area is titled "Welcome to UR Private Advisor" and contains a "Log In" form. The form has two input fields: "Email" and "Password". Both fields have red error messages above them: "Can't be empty or wrong format!" for the email field and "Can't be empty or wrong password!" for the password field. Below the fields is a yellow "Login" button. At the bottom of the form, there is a link that says "Do not have a account? Sign Up".

Robustness:

Input action - User tries to login with incorrect credentials that do not meet the requirements.

Output action - The systems detects the invalid entries and presents error messages.

The screenshot shows a web browser window with the address bar displaying `localhost/CS372Project/Login.html`. The page header features the University of Regina logo and a dark green navigation bar. Below the header, the text "Welcome to UR Private Advisor" is displayed. The main content area is a gray box titled "Log In" containing two input fields. The "Email" field has the text "mail.com" and a red error message "Can't be empty or wrong format!". The "Password" field has six dots and a red error message "Can't be empty or wrong password!". Below the fields is an orange "Login" button and a link that says "Do not have a account? Sign Up".

University of Regina

Welcome to UR Private Advisor

Log In

Can't be empty or wrong format!

Email

Can't be empty or wrong password!

Password

Login

Do not have a account? [Sign Up](#)

Robustness:

Input action - User tries to login using a valid email address, but the email is unregistered for this application.



Welcome to UR Private Advisor

Log In

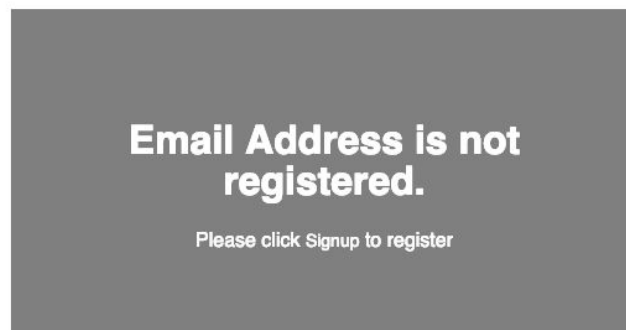
Email

Password

Login

Do not have a account? [Sign Up](#)

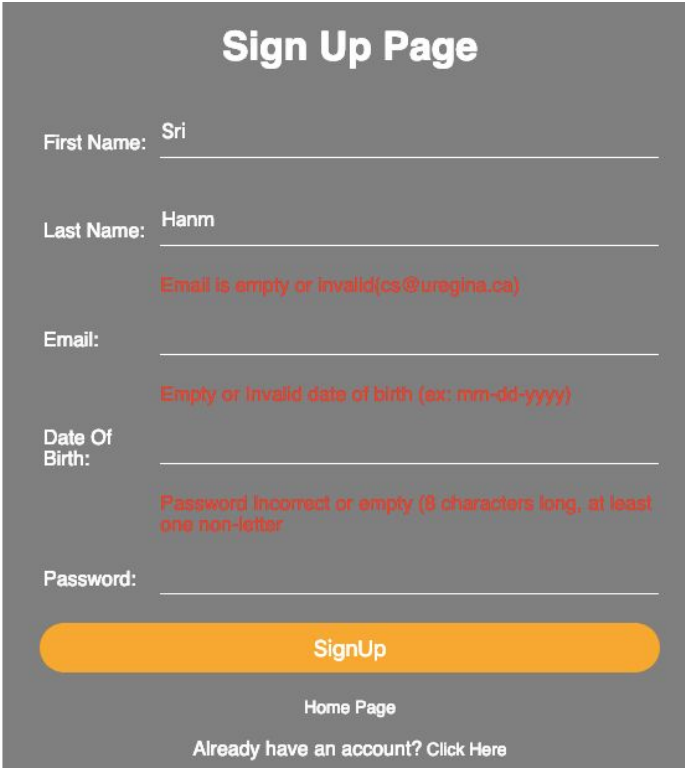
Output action: The application pops up the message that the user account does not exist in the application and advises them to go to the registration page, to access the application.



Robustness:

Input action - User tries to sign up using invalid or missing information.

Output action - The systems presents error messages.



Sign Up Page

First Name:

Last Name:

Email:

Email is empty or invalid(cs@uregina.ca)

Date Of Birth:

Empty or Invalid date of birth (ex: mm-dd-yyyy)

Password:

Password incorrect or empty (5 characters long, at least one non-letter)

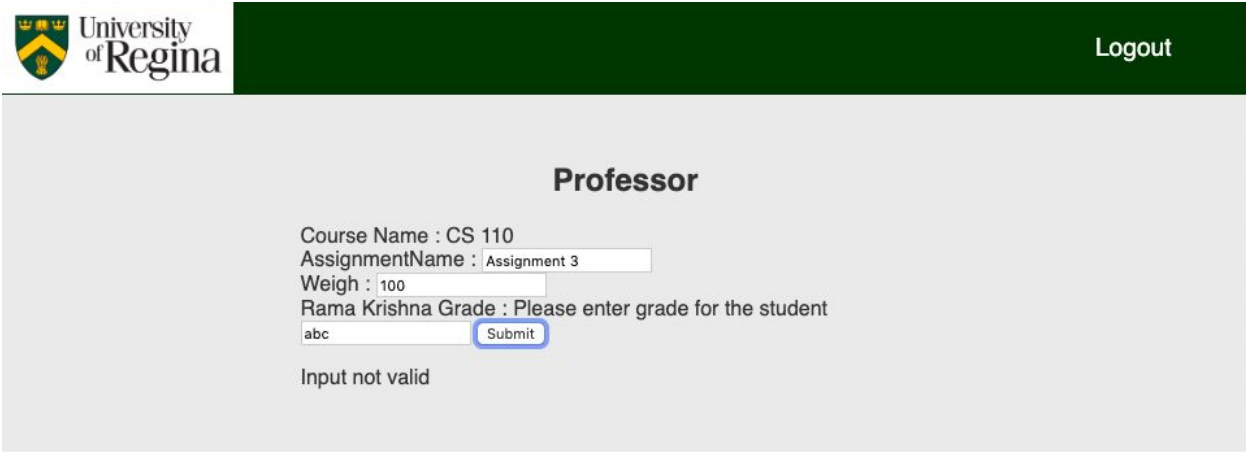
[Home Page](#)


[Already have an account? Click Here](#)

Robustness:

Input action: The Professor tries to enter invalid grades.

Output action: The system checks for validity and gives error message saying the input is not valid.



 [Logout](#)

Professor

Course Name : CS 110

AssignmentName :

Weigh :

Rama Krishna Grade : Please enter grade for the student

Input not valid

Performance testing:

We performed the performance testing for our website using the Pingdom tool (<https://tools.pingdom.com/>). Below are the test results for all our web pages.

Main Page: It took about 1.9 seconds to load the main page.

Unofficial Transcript: It took about 1.1 seconds to load the page.

Search for your program: It took about 1.9 seconds to load the page.

Current Status: It took about 1.0 seconds to load the page.

Professor's PutMarksIntoDatabase Page: It took 84 ms to load the page.

Division of Group Labour

Coding:

- Frontend: Sri Han, Logan Slater
- Business Logic Layer: Joel Lesko, Logan Slater, Brayden Hurl, Sri Han
- Backend: Joel Lesko

Report:

- Part 1: Joel Lesko
- Part 2: Joel Lesko
- Part 3: Joel Lesko, Brayden Hurl
- Part 4: Joel Lesko
- Part 5: Sri Han, Joel Lesko
- Part 6: Sri Han
- Proofreading/editing: Logan Slater