



Course Project Report for Undergraduate Students

Course Type: Subject Elective

Course Name: Internet Application Development
互联网应用开发(全英)

Course Code: 60080049

Instructor: 何明昕, HE Mingxin

《Web-Based Application: Select Courses》

Name 黄稻浪

Student ID 2016052948

College International

Major CST

Email 949837845@qq.com

Submit Date: 2018.12.20

Table of Contents

1. Project Description	3
1.1 Problem Statement	3
1.2 Design Tools.....	3
1.2.1 Vaadin	3
1.2.2 MyBatis	4
1.2.3 Jetty	5
1.3 Database	6
1.4 Instructions	7
2. Project Design	8
2.1 UI Part.....	8
2.2 Logic Part	11
3. Snapshots	15
3.1 Components	15
3.2 Overall Process	17

1. Project Description

1.1 Problem Statement

Design a web-based partial application with Vaadin + MyBatis on Teaching Administration Management, and mainly implement Select-Courses with necessary helper features (view selectable courses and selected courses, adjust selected courses etc.), considering the following constraints: prerequisite courses, credit limit (no more than 21 credits in one semester), and no time conflicts for selected courses.

1.2 Design Tools

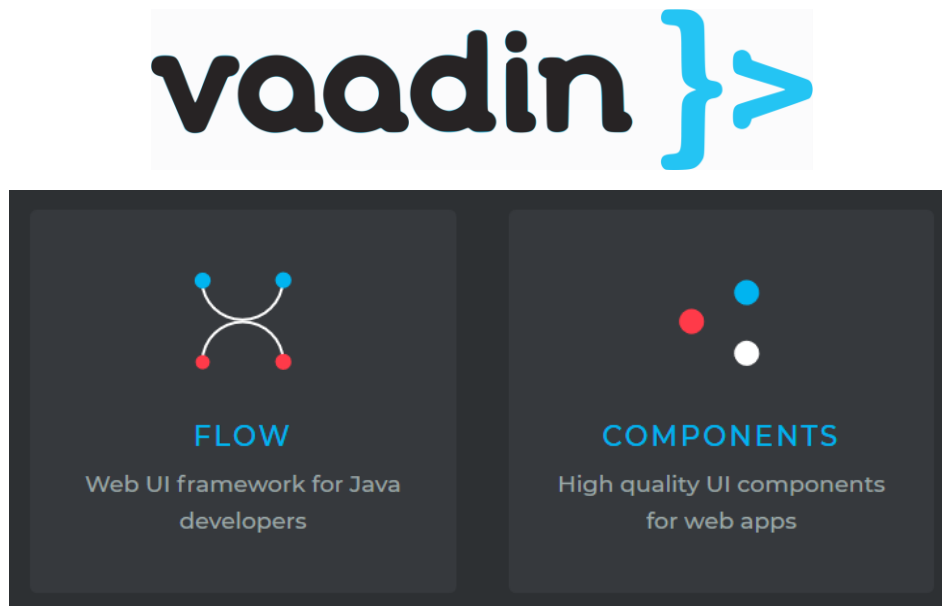
1.2.1 Vaadin

Vaadin is an open-source platform for web application development. The Vaadin platform includes a set of web components, a Java web framework, and a set of tools and application starters. Its flagship product, Vaadin Flow (previously Vaadin Framework) allows the implementation of HTML5 web user interfaces using the Java Programming Language.

Vaadin's components are a comprehensive set of Web Components for application developers. The components can be used in web documents (without frameworks) and web frameworks compatible with Web Components. These components are the core of Vaadin Flow, a Java web framework that offers a Java API on top the each Vaadin

component.

Vaadin Flow is a web framework for building web applications and websites. Vaadin Flow programming model is similar to Vaadin Framework's–It uses Java as the programming language for creating web content. Vaadin Flow features a server-side architecture which means that most of the logic runs on the server. On the client-side, Vaadin Flow is built on top of Web Component standards.



1.2.2 MyBatis

MyBatis is an excellent persistence layer framework that supports custom SQL, stored procedures, and advanced mapping. MyBatis avoids almost all JDBC code and manually sets parameters and gets result sets. MyBatis can use simple XML or annotations to configure and map native information, mapping interfaces and Java POJOs (Plain Old Java Objects) to records in the database.

Unlike ORM frameworks, MyBatis does not map Java objects to database tables but Java methods to SQL statements.

MyBatis lets you use all your database functionality like stored procedures, views, queries of any complexity and vendor proprietary features. It is often a good choice for legacy or de-normalized databases or to obtain full control of SQL execution. It simplifies coding compared to JDBC. SQL statements are executed with a single line.

MyBatis provides a mapping engine that maps SQL results to object trees in a declarative way. SQL statements can be built dynamically by using a built-in language with XML-like syntax or with Apache Velocity using the Velocity integration plugin.



1.2.3 Jetty












Eclipse Jetty is a Java HTTP (Web) server and Java Servlet container. While Web Servers are usually associated with serving documents to people, Jetty is now often used for machine to machine communications, usually within larger software frameworks. Jetty is developed as a free and open source project as part of the Eclipse

Foundation. Jetty supports the latest Java Servlet API (with JSP support) as well as protocols HTTP/2 and WebSocket.



1.3 Database

We used university database for the entire project. It contains eleven tables; the scope of each table is different but connect with each other. The structure of the entire database is as follows:

-  advisor
-  classroom
-  course
-  department
-  instructor
-  prereq
-  section
-  student
-  takes
-  teaches
-  time_slot

In this project, we mainly use five tables for the time being. They are course, prereq, section, takes and student. Among them, course is used to display all the courses that can be selected. Takes to show all the courses selected by students. The role of prereq is to check whether some courses require pre-requisite courses. Section to get the class time, and student to get the student's information.

At present, the problem found in this database is that some

courses in the course cannot find the corresponding course details in the section, which leads to some courses cannot be selected.

1.4 Instructions

Because the entire system is not online, so you have to use Jetty to test in your IDE. Details as follows:

1. Use your IDE like Eclipse or IntelliJ IDEA, and import this project as Maven project.
2. You can find MyBatis config file named “mybatis.xml” in resources folder. Then you have to set the url, username and password to your own university database. After doing this, you can successfully fetch the data from your database.
3. Run the Maven goal jetty: run in your IDE. Then go to “localhost:8080/login” to start this system.

2. Project Design

2.1 UI Part

Because I used Vaadin flow to design the entire UI, so I didn't use any traditional html or css to build the user interface. I used some Vaadin components in pure JAVA expressions, and it is very simple to build it.

However, there are many drawbacks to using Vaadin to build a UI. The limitations of Vaadin are very high. During my use, I have encountered the following problems:

1. You want to implement a function without knowing whether the control is packaged according to your idea.
2. Multi-layer control nesting Some mechanisms are written to die and cannot be controlled.
3. The choice of framework is single.
4. You want to abandon the framework, but find problems that cannot be achieved from the bottom.

The next few tables are some of the main problems and solutions I have encountered, as well as examples of some of the code. However, I omitted some less important problems.

1. Center the components

Problem	I want to center the components of the login interface.
---------	---

Solution	First add all components to a verticalLayout, then set the AlignItems property of the verticalLayout to center.
Code	<pre>horizontal.add(login); horizontal.add(forget); layout.add(horizontal); layout.setAlignItems(Alignment.CENTER);</pre>

2. Jump Page

Problem	I want to switch to the main page after clicking the login button.
Solution	Use Vaadin's route navigation function to complete the jump.
Code	<pre>login.addClickListener(e-> { login.getUI().ifPresent(ui -> ui.navigate("main")); });</pre>

3. Set Styles

Problem	I want to set the style of the main login interface to make the overall login interface more beautiful.
Solution	Although Vaadin has the ability to import style sheets, since there are not many styles required for this project, you can use Vaadin's built-in functions for css style design.
Code	<pre>title.getStyle().set("font-size", "30px"); title.getStyle().set("color", "grey"); forget.getStyle().set("margin-top", "10px"); forget.getStyle().set("margin-left", "25px"); forget.getStyle().set("color", "grey");</pre>

4. Tabs binding

Problem	Since I designed three tabs to help me switch views without switching URLs, how do I set up the interface so that different tables can be updated in real time?
Solution	A clever way can be used. By binding the tabs to their respective tables, and then clicking on different tabs, set the other components to be invisible, so that when the tab is toggled, the corresponding components can be seen separately.
Code	<pre>// set the Map for corresponding tabs and grids Map<Tab, Component> tabsToPages = new HashMap<>(); tabsToPages.put(tab1, all_course); tabsToPages.put(tab2, selected_course); tabsToPages.put(tab3, completed_course); Tabs tabs = new Tabs(tab1, tab2, tab3); Set<Component> pagesShown = Stream.of(all_course) .collect(Collectors.toSet()); // add a listener for the grid change tabs.addSelectedChangeListener(event -> { pagesShown.forEach(page -> page.setVisible(false)); pagesShown.clear(); Component selectedPage = tabsToPages.get(tabs.getSelectedTab()); selectedPage.setVisible(true); pagesShown.add(selectedPage); });</pre>

5. Select Button

Problem	I want to add an elective button after each line of the course.
Solution	Add a column to the grid and the column item is a button that can be rendered so that the button is dynamically added to each row of data.
Code	<pre>selected_course.addColumn(Takes::getCourse_id).setHeader("Course ID"); selected_course.addColumn(Takes::getSec_id).setHeader("Sec ID"); selected_course.addColumn(Takes::getSemester).setHeader("Semester"); selected_course.addColumn(Takes::getYear).setHeader("Year"); selected_course.addColumn(new NativeButtonRenderer<>("Cancel", item -> {</pre>

6. Completed Course

Problem	I want to let the user see the courses and grades that I have already learned before, so that students can check them.
---------	--

Solution	Create a new grid to display all completed courses, which are available in the database takes table.
Code	<pre> completed_course.addColumn(Takes::getCourse_id).setHeader("Course ID"); completed_course.addColumn(Takes::getSec_id).setHeader("Sec ID"); completed_course.addColumn(Takes::getSemester).setHeader("Semester"); completed_course.addColumn(Takes::getYear).setHeader("Year"); completed_course.addColumn(Takes::getGrade).setHeader("Grade"); completed_course.setVisible(false); update_completed_course(); // Update the grid of completed courses public void update_completed_course() { completed_course.setItems(DataService.getTakes(LoginView.get_id())); } </pre>

2.2 Logic Part

As for the logic part, we have to complete the database and UI interaction, as well as a series of restrictions. Therefore, the logic is very complicated and there are many problems to be solved. Therefore, only some of the most important issues have been selected for explanation. There are some issues left to explore in the source code.

1. Get Student ID

Problem	I want to get the student's id through the login interface and pass it to the database to get the student's data.
Solution	Get the student id through the getvalue function, and then pass it to the database to match the student's personal information.
Code	<pre> public static String get_name() { String id = userID.getValue(); return DataService.getStudent(id).get(0).getName(); } </pre>

2. Grid Binding

Problem	I want to get information about all the courses from the database and put it into the interface.
Solution	First create an XML mapping file, write a select sql statement in it, get all the course information in the course table. Then by creating a course entity class, and declaring the type of the grid as the entity class, and then through the data binding function of Vaadin, data binding can be achieved.
Code	<pre>all_course.addColumn(Course::getCourse_id).setHeader("Course ID"); all_course.addColumn(Course::getTitle).setHeader("Course Title"); all_course.addColumn(Course::getDept_name).setHeader("Department"); all_course.addColumn(Course::getCredits).setHeader("Credits");</pre>

3. Button Listener

Problem	I want to bind the select button. When I click the select button of a course, I can successfully write to the database and prompt the user to succeed or fail.
Solution	Vaadin's NativeButtonRenderer component can add a listener to it. When you click this button, you can pop up a dialog and write the data to the takes table of the database. Note that the grade option in the takes should be null.
Code	<pre>all_course.addColumn(new NativeButtonRenderer<>("Select", item -> { Dialog dialog = new Dialog(); dialog.setCloseOnEsc(false); dialog.setCloseOnOutsideClick(false); VerticalLayout dialog_vertical = new VerticalLayout(); HorizontalLayout dialog_horizontal = new HorizontalLayout(); Label message = new Label("Are you sure to choose this course?"); Button confirm = new Button("Confirm", event ->{ Course select = item; List<Section> section = DataService.getSection(select.getCourse_id()); if(time_conflict(section.get(0).getTime_slot_id()) == false && prereq_conflict(select.getCourse_id()) == false) { Takes take = new Takes(); take.setCourse_id(select.getCourse_id()); take.setID(LoginView.get_id()); take.setSec_id(section.get(0).getSec_id()); take.setSemester(section.get(0).getSemester()); take.setYear(section.get(0).getYear()); take.setGrade(null);</pre>

```

try {
    DataService.insert(take);
    update_selected_course();
} catch (Exception e1) {
    // TODO Auto-generated catch block
    e1.printStackTrace();
}
credit.setText("You have already selected "+get_selected_credits() +
    " credits. Your remaining credits for this semester are "+ (21-get_selected_credits())+ " credits.");
dialog.close();

// add a dialog to inform successfully selected
Dialog confirm_info = new Dialog();
VerticalLayout dialog_vertical2 = new VerticalLayout();
dialog_vertical2.setAlignItems(Alignment.CENTER);
Label message1 = new Label("Successfully Selected !");
Button confirm1 = new Button("OK", event1 ->{
    confirm_info.close();
});
dialog_vertical2.add(message1, confirm1);

confirm_info.add(dialog_vertical2);
confirm_info.open();
update_selected_course();
}
else {
    dialog.close();
    // add a dialog to inform time has conflict
    Dialog repeat = new Dialog();
    VerticalLayout dialog_vertical3 = new VerticalLayout();
    dialog_vertical3.setAlignItems(Alignment.CENTER);
    Label message2 = new Label("Time Conflict or Prerequisite Course Required !");
    Button confirm2 = new Button("OK", event1 ->{
        repeat.close();
    });
    dialog_vertical3.add(message2, confirm2);
    repeat.add(dialog_vertical3);
    repeat.open();
}

});
Button cancel = new Button("Cancel ", event->{
    dialog.close();
});
// ...

dialog_vertical.setAlignItems(Alignment.CENTER);
cancel.getStyle().set("margin-left", "20px");
dialog_horizontal.add(confirm, cancel);
dialog_vertical.add(message, dialog_horizontal);
dialog.add(dialog_vertical);
dialog.open();
}).setFlexGrow(0);
update_all_course();

```

4. Prerequisite Course

Problem	Set restrictions, if a course to be selected requires completion of another course, then you need to set the prerequisites course judge.
Solution	The course id is used to determine whether the course exists in the prereq table, and if so, check whether its prereq course has been completed by the student.

Code	<pre> public Boolean prereq_conflict(String course_id) { List<Takes> temp = new ArrayList<Takes>(); List<Takes> takes = DataService.getTakes(LoginView.get_id()); for (int i = 0; i < takes.size(); i++) { if(takes.get(i).getGrade() != null) { temp.add(takes.get(i)); } } List<Prereq> prereq = DataService.getPrereq(); for (int i = 0; i < prereq.size(); i++) { if(course_id.equals(prereq.get(i).getCourse_id())) { for (int j = 0; j < temp.size(); j++) { if(prereq.get(i).getPrereq_id().equals(temp.get(j).getCourse_id())) { return false; } } return true; } } return false; } </pre>
------	---

5. Time Conflict

Problem	If the time of one course conflicts with another selected course, you cannot choose this course.
Solution	First get the course id, pass it to the section table, compare the time slots of all selected courses, compare whether there is a conflict, if there is the same time slot, you cannot choose the class.
Code	<pre> public Boolean time_conflict(String time_slot) { List<Takes> temp = new ArrayList<Takes>(); List<Takes> takes = DataService.getTakes(LoginView.get_id()); for (int i = 0; i < takes.size(); i++) { if(takes.get(i).getGrade() == null) { temp.add(takes.get(i)); } } for (int i = 0; i < temp.size(); i++) { if (time_slot.equals(DataService.getSection(temp.get(i).getCourse_id()).get(0).getTime_slot_id())) { return true; } } return false; } </pre>

3. Snapshots

3.1 Components

3.1.1 Login Page

1) Overall Interface



The image shows the overall interface of the Course Selection System login page. At the top, there is the Jinan University logo, which consists of a circular emblem with a book and the year 1900, followed by the university's name in Chinese characters '暨南大學' and 'JINAN UNIVERSITY' in English. Below the logo, the title 'Course Selection System' is displayed in a large, dark gray font. Underneath the title, there are two input fields: 'User ID' and 'Password'. The 'User ID' field is a simple light gray rectangle. The 'Password' field is a light gray rectangle with a small eye icon on the right side to toggle visibility. Below the 'User ID' field, there is a blue button with a right-pointing arrow and the text 'Sign In'. To the right of the 'Sign In' button, there is a link that says 'Forget Password?'.

Figure 3.1.1.1 Overall Interface

2) User ID (TextField)



The image shows a close-up of the User ID input field. The label 'User ID' is written in blue text above the field. The field itself is a light gray rectangle containing the text '97711' followed by a vertical cursor line.

Figure 3.1.1.2 User ID Area

3) Password (PasswordField)

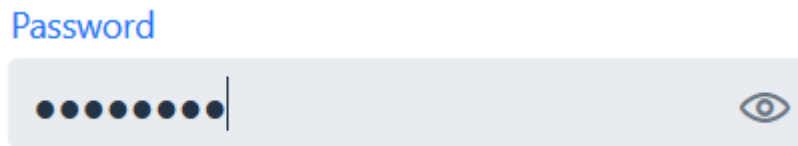


Figure 3.1.1.3 Password Area

4) Sign in Button and Forget Password Button



Figure 3.1.1.4 Sign in Button

3.1.2 Main Page

1) All the Courses Page

Hi Manber! Welcome to course selection system. Your department: Civil Eng. [Logout](#)

[All the Courses](#) [Selected Courses](#) [Completed Courses](#)

You have already selected 0 credits. Your remaining credits for this semester are 21 credits.

Course ID	Course Title	Department	Credits	
101	Diffusion and Phase Transformation	Mech. Eng.	3	Select
105	Image Processing	Astronomy	3	Select
123	Differential Equations	Mech. Eng.	3	Select
127	Thermodynamics	Geology	3	Select
130	Differential Geometry	Physics	3	Select
133	Antidisestablishmentarianism in Modern Ameri	Biology	4	Select
137	Manufacturing	Finance	3	Select
139	Number Theory	English	4	Select
158	Elastic Structures	Cybernetics	3	Select
169	Marine Mammals	Elec. Eng.	3	Select
190	Romantic Literature	Civil Eng.	3	Select
192	Drama	Languages	4	Select
195	Numerical Methods	Geology	4	Select

Figure 3.1.2.1 All the Courses Area

2) Selected Courses Page

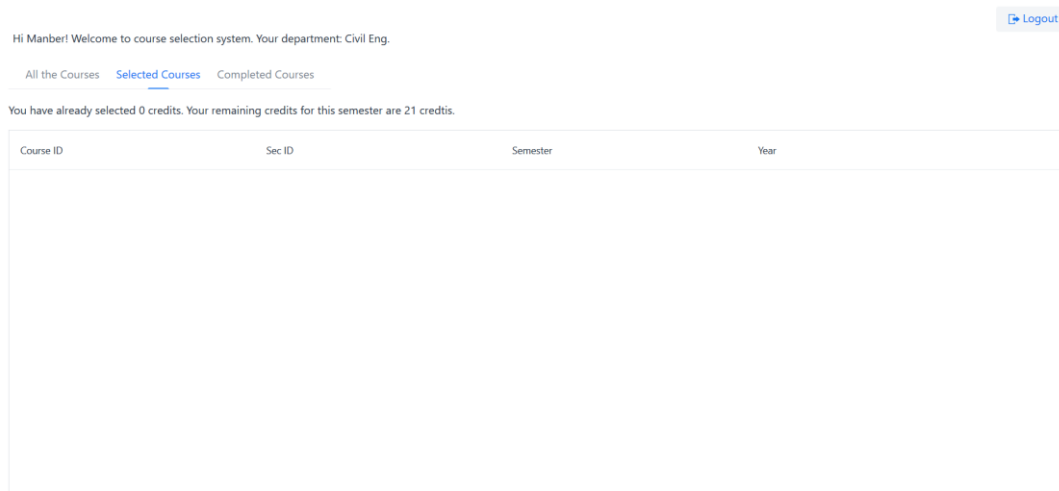


Figure 3.1.2.2 Selected Courses Area

3) Completed Courses Page

Hi Manber! Welcome to course selection system. Your department: Civil Eng.

All the Courses

Selected Courses

Completed Courses

You have already selected 0 credits. Your remaining credits for this semester are 21 credits.

Course ID	Sec ID	Semester	Year	Grade
319	1	Spring	2003	B+
362	1	Fall	2005	B+
493	1	Spring	2010	A-
571	1	Spring	2004	C+
642	1	Fall	2004	C-
663	1	Spring	2005	C+
696	1	Spring	2002	B
748	1	Fall	2003	A
802	1	Spring	2003	C+
959	1	Fall	2006	B+
962	1	Spring	2008	B-

Figure 3.1.2.3 Completed Courses Area

3.2 Overall Process


We selected the student with the ID 99711 to conduct the entire course of the course selection, including the success of the course selection and the failure of the course selection, and confirm whether the database data has been successfully written.

3.2.1 Login

Enter the ID, right now no password restraints.

User ID

Password




 Sign In [Forget Password?](#)

Figure 3.2.1.1 Enter User ID and Password

3.2.2 Check Student Information

We can see the information of student like name and his department.

Hi Deshpande! Welcome to course selection system. Your department: Pol. Sci.

Figure 3.2.2.1 Check the information of student 99711

3.2.3 Check Completed Course

We can click on Completed courses to view all completed courses.

Course ID	Sec ID	Semester	Year	Grade
200	2	Fall	2002	B
319	1	Spring	2003	A+
338	2	Spring	2006	B-
349	1	Spring	2008	C
400	1	Spring	2007	A+
415	1	Fall	2010	B-
489	1	Fall	2007	C
694	1	Fall	2002	C-
875	1	Spring	2005	C-

Figure 3.2.3.1 Check the Completed Courses

3.2.4 Select One Course

First, choose 192 and click select, it will alarm you whether to confirm, then you will receive a dialog to inform you successfully selected. And the remain credits information changed and you can see

the course in the selected course page. Besides, database has added this course to takes table.

You have already selected 0 credits. Your remaining credits for this semester are 21 credits.

Course ID	Course Title	Department	Credits	
127	Thermodynamics	Geology	3	Select
130	Differential Geometry	Mathematics	3	Select
133	Antidisestablishmentarianism in Mode	History	4	Select
137	Manufacturing	Engineering	3	Select
139	Number Theory	Mathematics	4	Select
158	Elastic Structures	Cybernetics	3	Select
169	Marine Mammals	Elec. Eng.	3	Select
190	Romantic Literature	Civil Eng.	3	Select
192	Drama	Languages	4	Select
195	Numerical Methods	Geology	4	Select
200	The Music of the Ramones	Accounting	4	Select
209	International Trade	Cybernetics	4	Select
224	International Finance	Athletics	3	Select

Figure 3.2.4.1 Confirm Dialog

You have already selected 4 credits. Your remaining credits for this semester are 17 credits.

Course ID	Course Title	Department	Credits	
127	Thermodynamics	Geology	3	Select
130	Differential Geometry	Mathematics	3	Select
133	Antidisestablishmentarianism in Mode	History	4	Select
137	Manufacturing	Engineering	3	Select
139	Number Theory	Mathematics	4	Select
158	Elastic Structures	Cybernetics	3	Select
169	Marine Mammals	Elec. Eng.	3	Select
190	Romantic Literature	Civil Eng.	3	Select
192	Drama	Languages	4	Select
195	Numerical Methods	Geology	4	Select
200	The Music of the Ramones	Accounting	4	Select
209	International Trade	Cybernetics	4	Select
224	International Finance	Athletics	3	Select

Figure 3.2.4.2 Prompt Dialog

You have already selected 4 credits. Your remaining credits for this semester are 17 credits.

Course ID	Sec ID	Semester	Year	
192	1	Fall	2002	Cancel

Figure 3.2.4.3 Selected Courses Page

ID	course_id	sec_id	semester	year	grade
99710	603	1	Fall	2003	B+
99710	795	1	Spring	2004	A-
99710	949	1	Fall	2007	B
99711	192	1	Fall	2002	(Null)
99711	200	2	Fall	2002	B
99711	319	1	Spring	2003	A+
99711	338	2	Spring	2006	B-

Figure 3.2.4.4 Database Records

3.2.5 Select Failed

Case 1 Time Conflict Restraints

First, choose 105, it will successfully be selected, then choose 242, it will fail, because 105 and 242 have time conflict.

You have already selected 7 credits. Your remaining credits for this semester are 14 credits.

Course ID	Course Title	Department	Credits	
236	Design and Analysis of Algorithms	Mech. Eng.	3	Select
237	Surfing		3	Select
238	The Music of Donovan		3	Select
239	The Music of the Ramones		4	Select
241	Biostatistics		3	Select
242	Rock and Roll	Marketing	3	Select
254	Security	Cybernetics	3	Select
258	Colloid and Surface Chemistry	Math	3	Select
265	Thermal Physics	Cybernetics	4	Select
267	Hydraulics	Physics	4	Select
270	Music of the 90s	Math	4	Select
272	Geology	Mech. Eng.	3	Select
274	Corporate Law	Comp. Sci.	4	Select

Time Conflict or Prerequisite Course Required !

OK

Figure 3.2.5.1 Prompt Dialog

course_id	sec_id	semester	year	building	room_number	time_slot_id
105	1	Fall	2009	Chandler	375	C
137	1	Spring	2002	Fairchild	145	I
158	1	Fall	2008	Whitman	434	F
158	2	Spring	2008	Taylor	812	D
169	1	Spring	2007	Gates	314	A
169	2	Fall	2002	Drown	757	L
192	1	Fall	2002	Polya	808	B
200	1	Spring	2007	Saucon	180	D
200	2	Fall	2002	Chandler	375	D
237	1	Spring	2008	Power	717	D
237	2	Fall	2009	Fairchild	145	J
242	1	Fall	2009	Fairchild	145	C

Figure 3.2.5.2 Time Slot Conflict

Case 2 Prerequisite Course Restraints

Choose 158, it will fail, because it requires you completed 408 first, but this student didn't finish 408 yet.

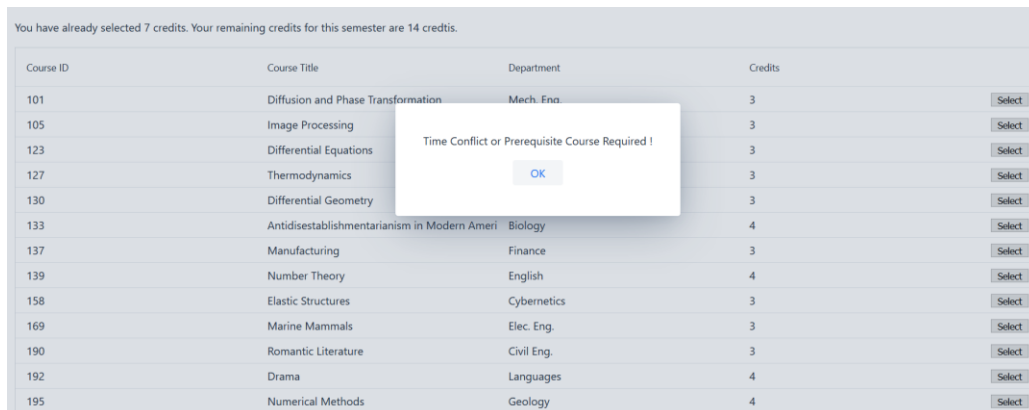


Figure 3.2.5.3 Prompt Dialog

course_id	prereq_id
133	852
158	408

Figure 3.2.5.4 Prerequisite Course

3.2.6 Cancel the Course

Change the page to selected courses, then click cancel of any course, then you will see the data be deleted in the database and view.

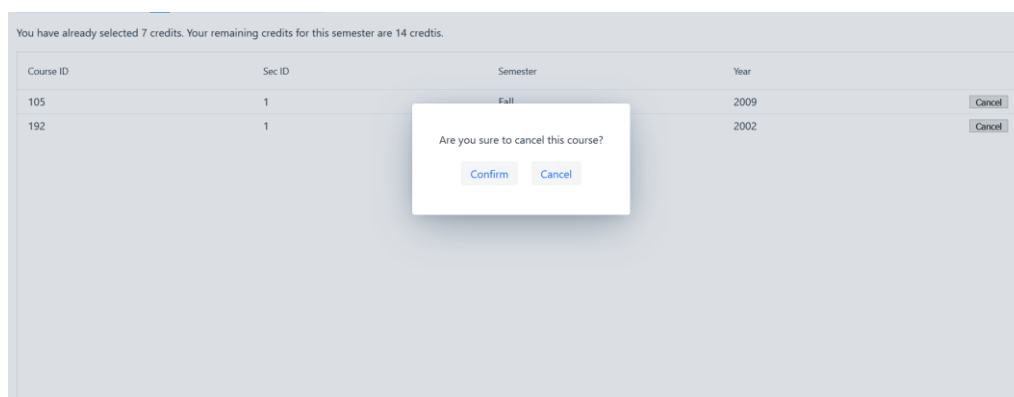


Figure 3.2.6.1 Check Dialog

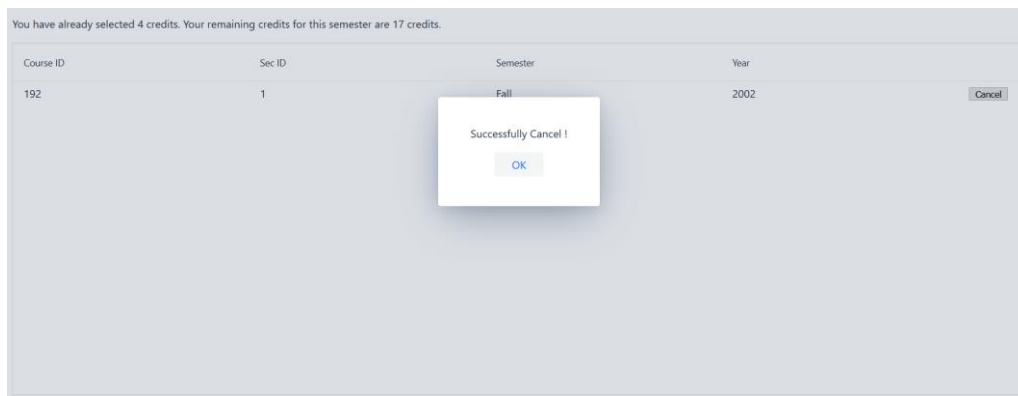


Figure 3.2.6.2 Inform Dialog

ID	course_id	sec_id	semester	year	grade
99710	603	1	Fall	2003	B+
99710	795	1	Spring	2004	A-
99710	949	1	Fall	2007	B
99711	200	2	Fall	2002	B
99711	319	1	Spring	2003	A+
99711	338	2	Spring	2006	B-
99711	349	1	Spring	2008	C
99711	400	1	Spring	2007	A+
99711	415	1	Fall	2010	B-
99711	489	1	Fall	2007	C
99711	694	1	Fall	2002	C-
99711	875	1	Spring	2005	C-

Figure 3.2.6.3 Data is Deleted