

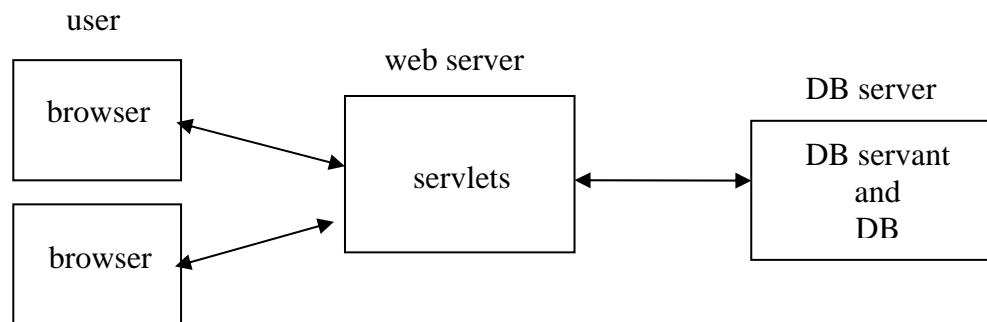
## COMPSCI334 Assignment 3

The objective of this assignment is to practice the use of Java servlet technology.

In this assignment, you are required to develop a simple web-based course enrolment system. The system consists of two main components, i.e. servlets on a web server and database servant programs on a database server. When the system is deployed, the web server and the DB server are expected to reside on different machines.

Users interact with the system using web browsers. Users must enrol in the system to get a UPI before they are allowed to enrol in any course. Users' requests are handled by a set of servlets. The database server hosts a database which records the information about the users and the courses. While handling users' requests, the servlets might need to access or update the information in the database. However, the servlets should not connect to the database directly. Instead, the servlets should interact with the database through a set of operations provided by the database servant programs running on the database server. The interaction between the servlets and the database servant programs should be implemented using Java RMI technology.

The interactions between various components are shown below:



### Part 1 (35 points)

Write the necessary servlets and DB servant programs which implement the function of enrolling users. The details are explained below.

#### DB Server

The zip file for this assignment contains a directory Asg3. Extract Asg3 from the zip file. There is a MS Access database file Asg3.mdb in Asg3. Asg3.mdb is used to store the data about students (i.e. users) and courses. There are several tables in Asg3.mdb. Some sample data are included in the database. A different set of data will be used by the marker. You can insert/delete data into/from the tables when you test your programs. However, you should NOT change (a) the design of the tables, or (b) the name of the tables, or (c) the fields' names of the tables. The tables in Asg3.mdb are as below:

**allcourses:** This table contains all courses and their prerequisites. Each row of the table is a (course, prerequisite of the course) pair. If a course has no prerequisite, the prerequisite field of the course is set to "non". If a course has more than one prerequisite, there are several rows for the course in the table. For example, the prerequisites for 320 are 220 and 225. Thus, (320, 220) and (320, 225) are both in the table.

**currentcourses:** This table records the courses that students currently enrol. Each row of the table is a (upi, courses) pair where upi is the upi of a student and courses is a course that the student currently enrolls. If a student enrolls in more than one course,

there are several rows with that student's upi in the table. For example, xye002 has enrolled in 220 and 225. Thus, there are two rows for xye002 in the table.

pastcourses: This table records the courses that students have completed. Each row of the table is a (upi, courses) pair where upi is the upi of a student and courses is the course that the student has completed. If a student has completed more than one course, there are several rows with that student's upi in the table. For example, xye002 has completed 101 and 105. Thus, there are two rows for xye002 in the table.

students: This table records the details of students who have enrolled in university. Each row of the table consists of four fields (upi, password, familyName, firstName) where upi is the upi of a student. The meanings of the other fields are obvious.

timetable: This table shows the lectures' time of each course. Each row of the table is a (course, day, time) tuple. There are five possible values for day field, i.e. Mon, Tue, Wed, Thu and Fri. The format for time is hour:minute where (a) hour and minute both consists of two digits and (b) hour varies between 00 and 23. It should be assumed that (a) each lecture lasts one hour, (b) a lecture starts either at the start of an hour, e.g. 11:00, or thirty minutes past an hour, e.g. 11:30, and, (c) a course can have any number of lectures a week.

All necessary programs which implement the functionalities of the DB servant should be placed in directory Asg3. You can figure out the operations that need to be provided by the DB servant programs while developing the servlet programs.

## Web Server

The zip file for this assignment contains a directory Asg. Extract Asg from the zip file. Place Asg under Tomcat's webapps directory. In Asg, there are two files which you might want to use. index.html is the starting page of the university's site. enrolment.html displays a form for a user to fill in when the user wants to enrol in the university. These are the only two static pages allowed in this assignment. The starting page of web site should look like below:



Figure 1

## Enrol in University

If a user clicks the enrol link in Figure 1, page enrolment.html is loaded and the following is shown to the user.



Address <http://localhost:8080/Asg/enrolment.html>

**Enrolment Page**

Please fill in the form below.

Family Name:

First Name:

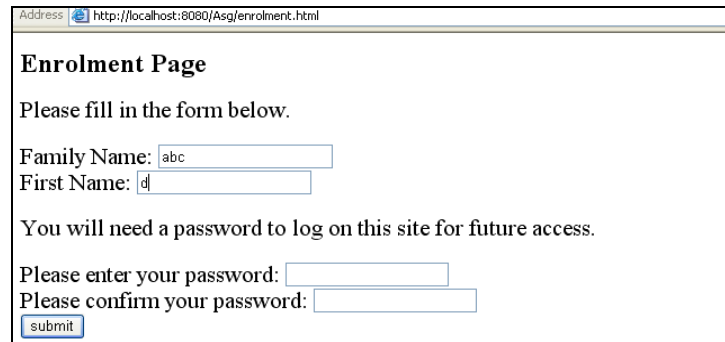
You will need a password to log on this site for future access.

Please enter your password:

Please confirm your password:

Figure 2

The servlet for handling a user's enrolment in the university should be called as ProcessEnrol.java. The servlet is called when the user submits the form in Figure 2. It should be assumed that (a) family name and first name consist of letters or space character " " only, (b) a user will enter non-empty family name and first name, and, (c) a user will only use a string consisting of letters and/or digits as his/her password. ProcessEnrol.java should check (a) whether the two passwords are identical, and (b) whether the password fields are empty. If any of these cases occurs, an error message should be given. For example, if the user's inputs are as Figure 3(a), Figure 3(b) should be shown.



Address: http://localhost:8080/Asg/enrolment.html

### Enrolment Page

Please fill in the form below.

Family Name:


First Name:

You will need a password to log on this site for future access.

Please enter your password:

Please confirm your password:

Figure 3(a)



Address: http://localhost:8080/Asg/servlet/ProcessEnrol?familyName=abc&firstName=d&password1=&password2=

### Enrolment Page

**Password cannot be empty. Please re-enter them.**

Family Name:

First Name:

You will need a password to log on this site for future access.

Please enter your password:

Please confirm your password:

Figure 3(b)

If the user's inputs are as Figure 4(a), Figure 4(b) should be shown.



Address: http://localhost:8080/Asg/servlet/ProcessEnrol?familyName=abc&firstName=d&password1=&password2=

### Enrolment Page

**Password cannot be empty. Please re-enter them.**

Family Name:

First Name:

You will need a password to log on this site for future access.

Please enter your password:

Please confirm your password:

Figure 4(a)

Address <http://localhost:8080/Asg/servlet/ProcessEnrol?familyName=abc&firstName=d&password1=a&password2=ss>

### Enrolment Page

**Passwords must be identical. Please re-enter them.**

Family Name:

First Name:

You will need a password to log on this site for future access.

Please enter your password:

Please confirm your password:

Figure 4(b)

Note: In Figure 3(b) and Figure 4(b), the password fields should be cleared while the name fields should display what the user already entered.  
If there are errors in both the name and password fields, you are only required to warn the user about one of the errors.

If the passwords are entered correctly as shown in Figure 5(a), then the page similar to Figure 5(b) should be shown.

Address <http://localhost:8080/Asg/enrolment.html>

### Enrolment Page

Please fill in the form below.

Family Name:

First Name:

You will need a password to log on this site for future access.

Please enter your password:

Please confirm your password:

Figure 5(a)

Address <http://localhost:8080/Asg/servlet/ProcessEnrol?familyName=abcd&firstName=efg&password1=abcd&password2=abcd>

Your enrolment has been completed successfully.  
Your UPI is eabc098  
You can enrol [here](#).

Figure 5(b)

From Figure 5(b), it can be seen that an UPI is generated for the user. The UPI of a student must be unique. The general rules for generating an UPI are as below:

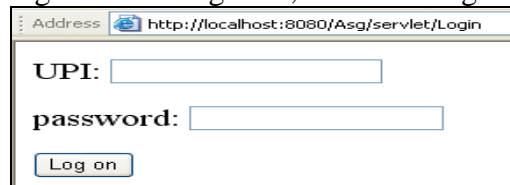
- An UPI consists of three parts which are denoted as Part1Part2Part3.
- Part1 is the first letter of the first name of the user. It must be a lower case letter.
- Part2 is the first three letters of the family name of the user. All letters must be lower case letters. If a user's family name consists of less than three letters, e.g. ye, then Part2 is equal to the user's family name.
- Part3 is a three digits number varying between 001 and 999.
- Two different users might have identical Part1 and Part2 in their UPIs. If this is the case, Part3 of their UPIs must be different.


After a student enrolls successfully, the UPI, password and the name of the student should be added to the students table in the DB.

### Part 2 (10 points)

Write the necessary servlets and DB servant programs which implement the functions of logging in users and displaying the home page of the users.

When a user clicks the log in link in Figure 1, the following should be displayed:



Address  http://localhost:8080/Asg/servlet/Login

UPI:

password:

The servlet for handling a user's log in should be called as Login.java. If a password does not match an UPI, the user should be notified as below:



Address  http://localhost:8080/Asg/servlet/Login?upi=&password=8B1=Log+on

**Password is incorrect.**

UPI:

password:

If a user, say xinfeng ye, logs in successfully, the following should be displayed (Note: The name of the user should appear in the welcome message.).



Address  http://localhost:8080/Asg/servlet/Login?upi=xie002&password=xie002&B1=Log+on

**Welcome xinfeng ye**

You can list all courses [here](#).

You can list the courses that you have passed [here](#).

You can list the courses that you have enrolled [here](#).

You can enrol [here](#).

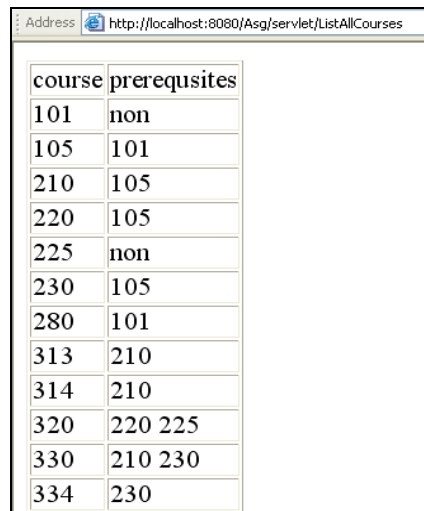
[Logout](#)

Figure 6

### Part 3 (15 points)

Write the necessary servlets and DB servant programs which implement the functions of (a) listing all courses offered by the university, (b) listing all courses that a user has passed, and (c) logging out a user.

When a user clicks the “list all courses” link in Figure 6, a page similar to Figure 7 should be displayed. The servlet which handles the click should be ListAllCourses.java. The information in the page should be obtained from table allcourses in the database. It should be noted that Figure 7 only shows a portion of the complete page to be displayed. That is, the complete page should include all the courses in table allcourses.

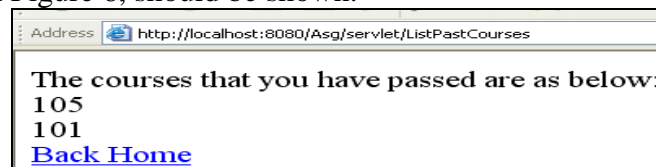


course	prerequisites
101	non
105	101
210	105
220	105
225	non
230	105
280	101
313	210
314	210
320	220 225
330	210 230
334	230

Figure 7

If a user clicks the “list the courses that you have passed” link in Figure 6, a page similar to Figure 8 should be shown. The servlet which handles the click should be ListPastCourses.java. The information in the page should come from table pastcourses in database. The courses listed in the page should be sorted into descending order. Figure 8 shows the output for user xye002.

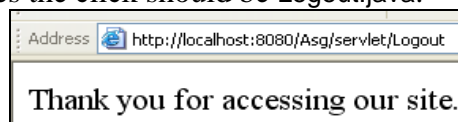
In Figure 8, when the user clicks the “Back Home” link, the home page of the user, i.e. the one shown in Figure 6, should be shown.



The courses that you have passed are as below:	
105	
101	
<a href="#">Back Home</a>	

Figure 8

When a user clicks the “Logout” link in Figure 6, a page below should be displayed. The servlet which handles the click should be Logout.java.



Thank you for accessing our site.
-----------------------------------

#### Part 4 (15 points)

Write the necessary servlets and DB servant programs which implement the function of listing all courses that a user has enrolled.

When a user, say xye002, clicks the “list the courses that you have enrolled” link in Figure 6, a page similar to Figure 9 should be displayed. The servlet which handles the click should be ListEnrolledCourses.java. The page not only shows the courses that the student has enrolled, it also shows the time table of the lectures of the enrolled courses. The items in the time table should be sorted according to day and time. The information in the page should be obtained from table currentcourses and timetable in the database.

Address <http://localhost:8080/Asg/servlet/ListEnrolledCourses>

The courses that you have enrolled are as below:

course	Times
220	Tue 15:00 Thu 14:00 Fri 15:30
225	Mon 08:00 Mon 09:00 Mon 11:00 Tue 14:00 Fri 13:00

[Back Home](#)

Figure 9

### Part 5 (15 points)

Write the necessary servlets and DB servant programs which implement the function of enrolling a course for a user.

When a user, say xye002, clicks the “You can enrol here” link in Figure 6, a page similar to Figure 10(a) should be displayed. The servlet which handles the click should be EnrolCourses.java. The combo box in Figure 10(a) should only include the courses which xye002 is eligible to enrol. A student is eligible to enrol in a course, if the condition below is true:

- The course does not have any prerequisite or the student has completed all the prerequisites of the course.  
and
- The student is not currently enrolled in the course.  
and
- The student has not previously completed the course.

Figure 10(b) shows the eligible courses for xye002. None of the stage 3 courses are shown, since xye002 has not passed the prerequisite(s) of any stage 3 courses. 220 and 225 are not shown, since xye002 is currently enrolled in these two courses. 101 and 105 are not shown, since xye002 has already completed these two courses.

Address <http://localhost:8080/Asg/servlet/EnrolCourses>

Please choose a course

Figure 10(a)

Address <http://localhost:8080/Asg/servlet/EnrolCourses>

Please choose a course

230  
280

Figure 10(b)

When the user clicks the “Enrol” button, servlet ProcessEnrolCourse.java should be used to handle the user’s request. Your program should check whether the course that the user wants to enrol has time table clash with the courses that the user has already enrolled. If there are time table clashes, the user should be notified. The time table of the courses involved in the clash should be displayed. Figure 11(a) shows the result when xye002 tries to enrol in 210. It can be seen that 210 clashes with both 220 and 225. Figure 11(b) shows the result when xye002 tries to enrol in 280. It can be seen that 280 only clashes with 225. Thus, only the time table of 225 and 280 are shown in Figure 11(b).

Address <http://localhost:8080/Asg/servlet/ProcessEnrolCourse?course=210&B1=Enrol>

210 clashes with some course(s). Please see the details below:

course	Times
220	Tue 15:00 Thu 14:00 Fri 15:30
225	Mon 08:00 Mon 09:00 Mon 11:00 Tue 14:00 Fri 13:00
210	Tue 14:30 Thu 14:30 Fri 14:30

[Back Home](#)

Figure 11(a)

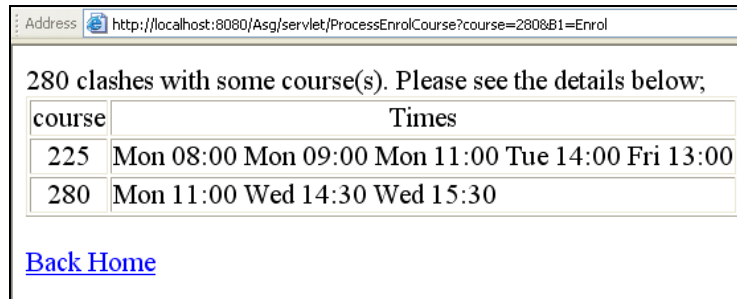


Figure 11(b)

An enrolment attempt is successful if the course that the student tries to enrol does not have any time table clash with other courses. Figure 12 shows the resulting page when xye002 tries to enrol in 230.



Figure 12

### Part 6 (10 points)

Improve the servlets that you have written in previous parts to enforce the following behaviour:

If a user has not logged in the system and tries to access a servlet directly by typing the name of the servlet, e.g. `http://localhost:8080/Asg/servlet/ListPastCourses`, then the starting page of the site shown in Figure 1 should be displayed.

### Report

You should write a report which covers the following issues:

- The parts that you have completed.
- Detailed instructions on running your program. You should include a batch file for running the database servant program(s).

### Submission

You MUST thoroughly test your programs in Tamaki lab before you submit your assignment. You should pack your programs into file `Asg3.zip` using a program like WinZip. You should put directory `Asg3` and `Asg` in `Asg3.zip`. You should make sure that the directory structure is preserved. The file which contains your report should also be included in `Asg3.zip`. Name your report file as “Report.doc”. You should submit file `Asg3.zip` through the assignment drop box.

**Total Points: 100 Points**

**Due Date: 23:59, 14<sup>th</sup> May 2008, sharp.**

**Electronic submission through the assignment drop box only.**

**No email submission will be accepted.**

### Appendix

In this assignment, all servlets’ URLs should have prefix “localhost:8080/Asg/servlet”. On Tomcat, this is what you should do in order to have a prefix



“localhost:8080/Asg/servlet” when referring to each servlet. Place directory Asg in the zip file for this assignment under webapps directory of Tomcat. The structure of Asg directory should be as below:



You should put all your servlets, i.e. the source code and the class files, in directory classes. Directory Asg3 which holds the database and the DB servant programs can be set up anywhere on your drive. I suggest you put Asg3 at a location outside your Tomcat installation to avoid confusion. When you submit your assignment, your ZIP file should include directories Asg3 and Asg.