**Table of Contents**

1. Application Overview
2. System Processes (use case diagram/description) - Done
3. System Flow (activity diagram) – Done
4. Object model (class diagrams)
5. User interface mockups (menus)
6. Data structures (text file format)
7. Test cases

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Changes** | **Version** |
| Kyle Hinsz | 05/13/14 | Initial Draft | 1.0 |
| Shamima Huq | 05/17/2014 | Overview, Test Cases, Data Structures, TOC, Revision history | 2.0 |

**Technical Design**

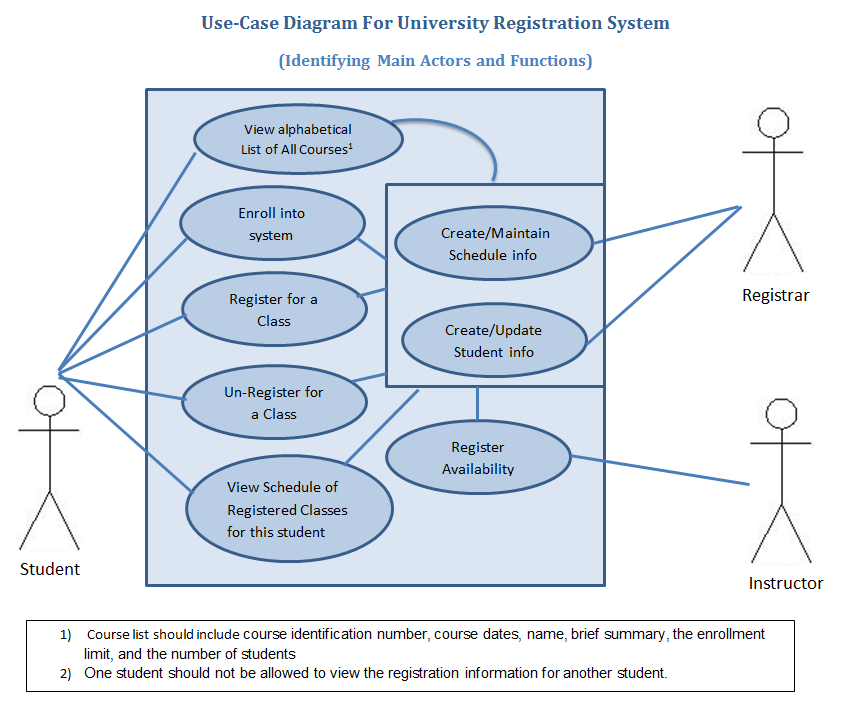
**1. Application Overview**

The application is a student registration system that displays an alphabetically ordered list of courses available for registration. The initial course list is a text file that includes the course identification number, course dates, name, brief summary, the enrollment limit, and the number of students already enrolled.

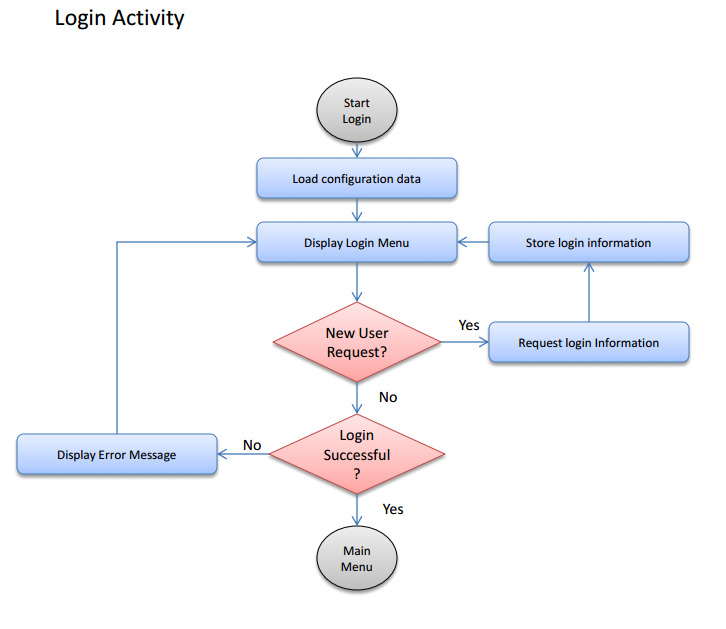
The program shows current registration numbers and available seats for each course. As students register/unregister the course’s counter of currently registered students will be adjusted. After a student successfully registers, the registration information is stored in a file and the system displays a list of classes for which he/she is currently registered.

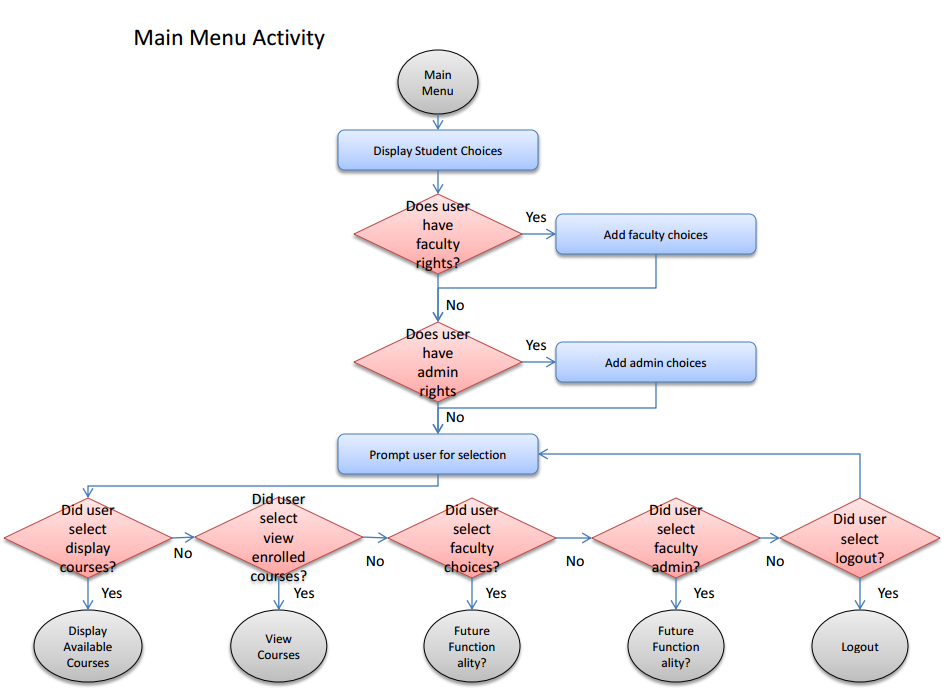
The system does not allow a student to register for a course beyond its maximum student capacity. This number is assigned in the initial text file as input. Also, before a student is able to un-register from a course, the system checks to make sure he/she was actually registered. Additionally, the system does not allow one student to view the registration information for another student.

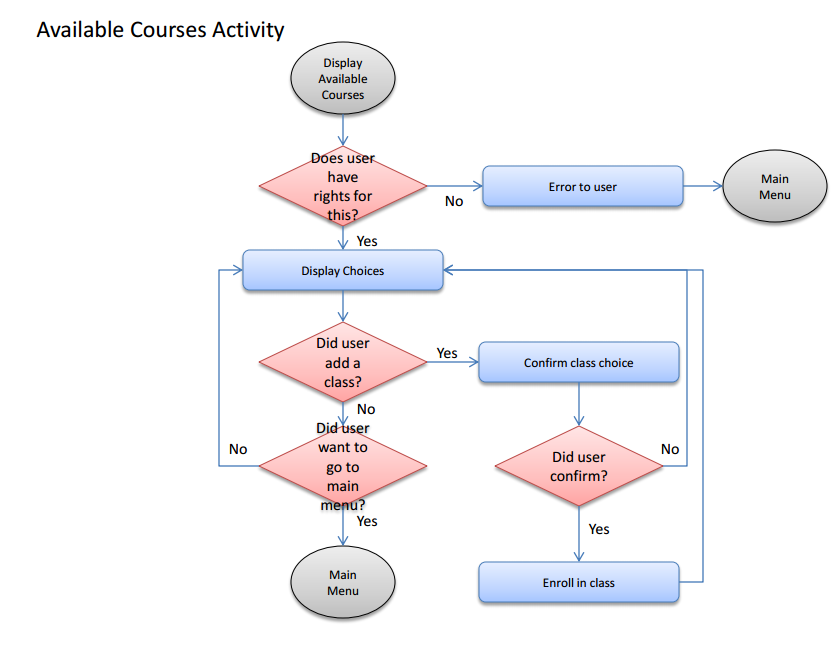
**2. System Process**

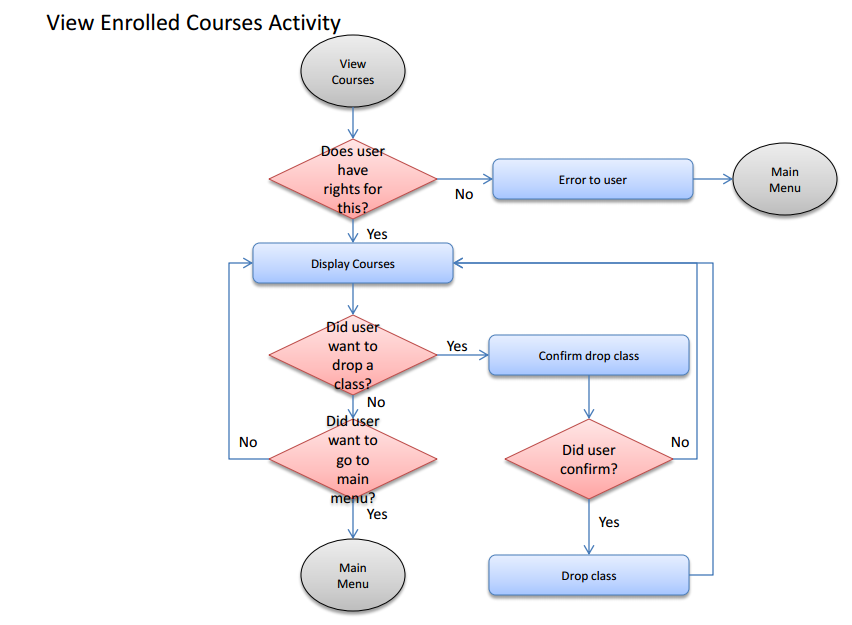


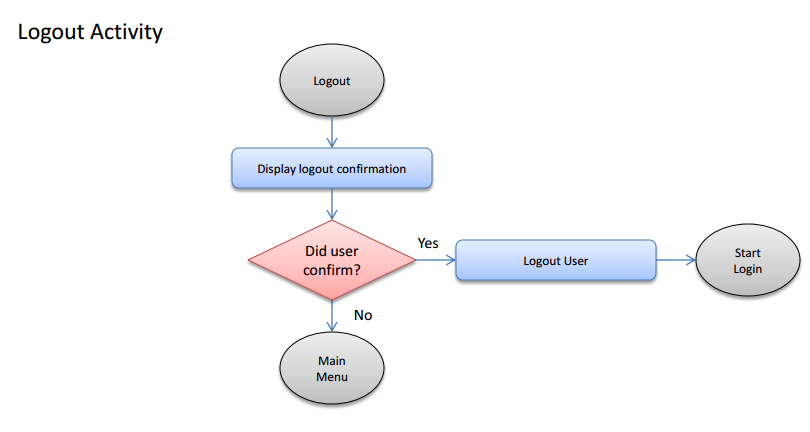
**3. System Flow**











**4. Object Model**

**4.1 Main Function**

**4.2 Course Class**

|  |
| --- |
|  |
| Variables  Identification Number : String  Start Date : Date  End Date : Date  Name : String  Summary : String  Enrollment Limit : Integer  Number of students Enrolled : Integer |
| Constructors  Course(String identificationNumber, Date startDate, Date endDate, String name, String summary, int enrollmentLimit, int studentsEnrolled) |
| Methods  String getIdentificationNumber() : Returns the course identification number  Date getStartDate() : Returns the course start date  Date getEndDate() : Returns the course end date  String getName() : Returns the course name  String getSummary() : Returns the course brief summary  Int getEnrollmentLimit() : Returns the course enrollment limit  Int getStudentsEnrolled() : Returns the current number of students enrolled in the course  void setIdentificationNumber(String identificationNumber) : sets the course identification  number  void setStartDate(Date startDate) : sets the course start date  void setEndDate(Date endDate) : sets the course end date  void setName(String name) : sets the course name  void setSummary(String summary) : sets the course summary  void setEnrollmentLimit(int enrollmentLimit) : sets the course enrollment limit  void setStudentsEnrolled(int studentsEnrolled) : sets the current number of students enrolled in  the course  void printCourse() : prints course information  void addStudent() : increments the current number of students enrolled by one  void removeStudent() : decrements the current number of student enrolled by one |

**4.3 Student Class**

**4.4 File I/O Utility Class**

|  |
| --- |
|  |
| Constants  Directory Path : String  Courses File : String  Students File : String |
| Constructors |
| Methods  Static ArrayList<Course> readCourseData() : Reads course data from the courses file and returns  the information in an array list of course objects  Static ArrayList<Student> readStudentData() : Reads student data from the students file and  returns the information in an array list of student objects  Static void saveCourseData(ArrayList<Course> courses) : Saves course information contained in  the courses ArrayList to a file  Static void saveStudentData(ArrayList<Student> students) : Saves course information contained  In the students ArrayList to a file |

**4.5 Login Utility Class**

|  |
| --- |
|  |
| Variables |
| Constructors |
| Methods  Static Student loginStudent(ArrayList<Student> studentList) : Prompts the user for a name and  password. Returns the student associated with the information. Returns null if login is  unsuccessful. |

**5. User interface mockups (menus),**

**6. Data structure**

**Data structure:**

**6.1 Course file data structure**

The course data file structure will be a comma separated text file list of fields in which one courses data is captured on each line. The data structure of each row will be:

courseIDnum, courseName, enrollmentLimit, studentsEnrolled, startDate, endDate, courseSummary

**6.2 Student file data structure**

studentUsrName, studentPW, studentEmail, studentID, studentName, studentDeprt, enrolledCourses, studentMaxCourses

**7. Test Cases**

Test cases will be prepared to evaluate functionality of the system as the application is developed. Initial few records have been added as students already in the system.

* Enroll into System
  + New user … create account
  + Existing user – (login pass/fail)
* Register for a Course
  + Register for a course – First time– (pass/fail)
  + Attempt to register for a course (already registered) – (pass/fail)
* Unregister for a Course
  + Unregister for a course (already registered) – (pass/fail)
  + Unregister for a course (not already registered) – (pass/fail)
* View Course List
  + View all registered courses – (pass/fail)
  + Check Alphabetic List of all Available Courses – (pass/fail)

The list below is the student text file input for the system

<Username>, <Password>, <Registered Course List>

Cecile2014, nm$rh&, MS201, SC203

SarahMAY13, Sr34%f^, CS101, MS104, SC305

Bill4456, 3654SRK@, MS403

Ly%3342, dsw@4567, MS405