CSC 415-01

Course Lion

Brian Worts

10/3/2019

Github: CourseLion -> https://github.com/brianworts/CourseLion

Overview (Option 3)

The goal of “Course Lion” is to add functionality to the existing Ruby application for enrolling students in computer science courses at TCNJ by adding additional constraints and improvements, thus making it more practical for use by the department.

The Problem

The College of New Jersey’s “PAWS” enrollment system is becoming increasingly outdated and in need of improvement. More and more manual work is required to enroll students by department heads due to a growing student population, which is leading to more students needing to get into classes. This option was chosen due to the desire to add additional functionality to the existing project and the need to create a better system than the current one.

Framework Overview

This program will be implemented via a Rails web based application that is linked to a SQL database of the data. These tables within the database will only be accessible by the system administrator, while the students have the ability to see only information pertaining to them from the tables. The algorithm will be implemented using Ruby.

Objectives & Planned functionality (Algorithm and Data Structures)

The objective of this project is to develop a system and website to improve upon the existing work done in “Assignment 1” to create a better enrollment system for computer science students at TCNJ. Although there will be many changes, one thing that is not being changed is the primary data structure for the algorithm. The current system uses hash tables as a means of storing all student and course objects when enrolling. This will remain since the primary actions of the algorithm are to lookup objects in these hash tables, which have a lookup time of O(1), via search “keys,” thus making them the perfect choice.

The first major improvement will be to improve the current algorithm in Ruby. In its current state, many students are not being enrolled in courses because the class is not reaching its minimum requirement despite there being other sections that are full. The updated algorithm will balance these courses out so that there is an even distribution of students which will allow for more students to enroll. The way the data is stored will also be changed to increase efficiency. Currently, lists of students enrolled in courses and a students course list are stored as strings, which makes printing them simple but searching them slow. These lists will be changed to arrays of string to make for faster searching and increase the overall speed. This data will also be stored in a SQL database as opposed to the current system of storing data in excel spreadsheets.

Another major improvement will be the movement of the system to a Rails web app and the creation of interfaces for students, to allow them to put in their course preferences and view the courses they were enrolled in once the program runs, and an interface for the computer science administrator to view the results of the course and to run the enrollment program.

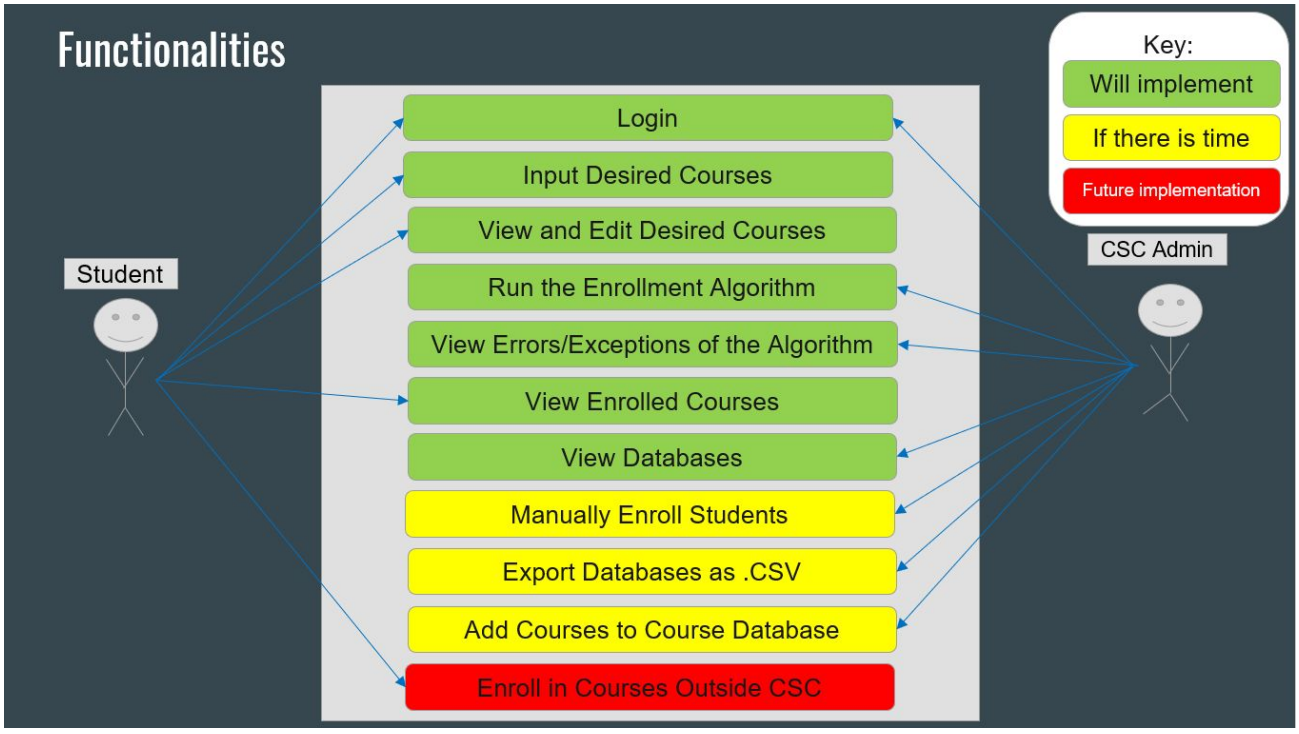
Lastly, there will also be additional information for student preferences and for course constraints. The student changes include checks involved when a student is selecting courses to verify if they have met the prerequisites and if the input is valid, thus reducing the need for checks later on and improving the enrollment algorithm's timing. Currently the system trusts that the user will input the truth regarding if they have met the prerequisites. In the new system, the student's course history will be stored and from that history the system will know whether the prerequisites have been met. If they have not met the prerequisites or there is some error regarding one of their inputs such as an incorrect course name, the student will be asked to enroll in another course instead of accepting the invalid input like it does now. Additionally, students will be given different priority when enrolling in a course based on their year in college. Seniors will receive top priority, juniors next, and so on. The additional course constraints will also include course times, so the existing algorithm will need to be improved to check for

timing conflicts between courses and output proper messages to elaborate when such conflicts occur.

Technologies and Concepts to Learn

The main program to learn will be Rails. I am familiar with both Ruby and SQL, but the challenge will come with integrating those programs into a Rails website. User interface design will also be an important concept to keep in mind as I build the website since a bad interface can cause a program to not be used at all no matter how good the algorithm is.

Use-Case Diagram



Timeline

#	Task	Start	End	Number of Days
	Total schedule time	10/3/2019	12/5/2019	71
1	Construct Website Framework	10/3/2019	10/7/2019	5
2	Algorithm Construction	10/7/2019	10/20/2019	14
3	SQL Construction & Integration	10/18/2019	10/27/2019	10
4	Get Feedback from Stakeholders	10/27/2019	10/31/2019	5
5	Algorithm Improvements & Integration	10/31/2019	11/10/2019	11
6	Website Imrpovement	11/10/2019	11/22/2019	13
7	Testing/Bug Fixing/Finalizing	11/22/2019	11/29/2019	8
8	Finalizing/Report	11/29/19	12/5/19	7