SCORE MyCourses – A Course Scheduling System

Team members: Will Crawford, Sabba Petri, Erik Steggall, Justin Lazaro, Benjamin Ross

**Requirements**

**Table of Contents**

**Change Record……………………………………………… 1**

**Overview…………………………………………………….. 1**

**Reference Documents………………………………………. 1**

**Definitions…………………………………………………... 2**

**Requirements……………………………………………….. 3**

* **[Functional: Algorithm, Database, User Interface, Backend]**

**Performance………………………………………………… 7**

* **[Usability, Constraints, Wish List, Coding Standards]**

**Preliminary User Interface……………………………………….9**

**Change Record:**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Changes/Additions | Responsible Person |
|  |  |  |  |
|  |  |  |  |

**Overview:**

The MyCourses scheduling system will allow program administrators, program managers, lecturers, and students to create, assign, detail and sign up for courses. This will be an end-to-end system that incorporates an algorithm that will automatically schedule courses based on a list of constraints. The program administrator will designate and list out all the possible courses, times, and locations. The program manager determines which courses will be offered and which lecturers will teach those courses during a particular quarter. The lecturer fills out his time constraints, and information about the course. The program administrator runs the algorithm which generates the final schedule of courses to be offered. Finally, the students signs up for the courses.

**Reference Documents:**

* **MyCourses Competition Description**

http://score-contest.org/2011/Projects.php#crnkovic

* **Python Language Reference**

http://docs.python.org/reference/

* **Django Framework Documentation**

http://docs.djangoproject.com/en/1.2/

* **Django Book, another Django reference**

http://www.djangobook.com/en/2.0/

* **Algorithm**

http://www.codeproject.com/KB/recipes/GaClassSchedule.aspx

**Definitions:**

1. **MyCourses** – The end-to-end course scheduling system
2. **PA** (Program Administrator) – The users who handle the entire MyCourses system
3. **PM** (Program Manager) – The users who handle individual department’s program and course offerings
4. **Quarter Course List** – This will be a list of courses offered in the particular quarter; every quarter will be chronologically unique, and will be independent from the other quarters
5. **MyPortal** – This will be the main user interface users will be interacting with via a web browser
6. **Users** – Users can be considered any users of the MyCourses system

**Requirements:**

* **Functional –** 
  + ***Algorithm*** - The main algorithm that will be implemented into S.C.O.R.E. will be a genetic algorithm developed by Mladen Jankovic.
    1. Because we’ve identified our scheduling problem as a NP-Complete problem, whereby there is no known polynomial-time procedure to find a solution, we need to use an algorithm that can handle a dynamic amount of constraints with proper speed and scale.

This genetic algorithm begins by creating “chromosomes,” or a representation of the variables to fit into our constraints that allows for genetic operations such as crossover and mutation.

* + 1. The algorithm then finds the fitness of these chromosomes, based on how well it fits the constraints. In this case, we have “slots” that are filled by “classes.”
    2. The algorithm can perform a crossover operation, whereby it takes two parents and randomly mixes them to create a child chromosome.
    3. The algorithm can perform a mutation operation, whereby it simply takes one “class” in a slot and moves it to another random slot.
    4. The algorithm runs by taking N pairs of parents from the current population in order to produce N new chromosomes by performing crossover operations on these parent pairs. It then replaces M chromosomes in the population with new chromosomes. It does not replace chromosomes with the highest fitness.
    5. The algorithm repeats the above step until there are chromosomes (at least one) with a satisfactory fitness level.
    6. Once complete, the courses will be generated and stored in the database and will be readily accessible to students.
  + ***Database*** – Our database system will be segregated into several tables. In the following list is the title of the database table and its various attributes.
    1. **Courses** –This will hold the table of all the courses offered at UCSC – it should reflect the course catalog.
       - Course Name
       - Course ID
       - Department
       - Description
       - Units
       - Prerequisites
    2. **Classrooms** – This table will list all of the classrooms on campus.
       - Building
       - Room Number
       - Class size
       - Preferred Department
    3. **Lecturers –**This will be used to provide lecturer’s teaching information for the quarter
       - ID
       - Lecturer Name
       - Department
    4. **Quarter Course List** –This will be used to provide which classes will be offered for the quarter.
       - Course ID
       - Quarter Course ID
       - Period ID
       - Class Time
       - Section
       - Lecturer
       - Lecturer’s Office Hours
       - TA ID
       - Building ID
       - Room ID
    5. **Person** – This table will hold information about each user.
       - ID
       - First Name
       - Middle Initial
       - Last Name
       - Suffix
    6. **Role** – This table will hold all possible roles a user can have.
       - ID
       - Role
    7. **PersonRole** – This table will define which role a user has.
       - Role ID
       - Person ID
  + ***User Interface*** – Each user who logs into the MyCourses page will have a customized user interface.
    1. Interface
       - The interface itself will be within the browser. It will be built with HTML, CSS and Javascript.
       - It will be compatible with all major browsers (Internet Explorer, Firefox, Safari, Chrome, Opera)
    2. Log in
       - Using their school email (without the @ucsc.edu) and password as their credentials.
         * Username: JohnDoe
         * Password: abcd1234
    3. Home Screen
       - When a user logs into the MyCourses system, the interface will be customized to their user role, determined by their supplied credentials.
         * i.e. Student UI will display different information from a PM UI
       - On the top of the screen will be a horizontal series of dropdowns that will let the user navigate around the site.
         * All users

Home – Home page to the site

Messages – Displays any messages that pertains to the user

* + - * + PA

Course System database – This will be a complete database separated into several tables

PA will populate these tables

Manage Program Manager – PA’s can view the progress and status of the PM’s work

* + 1. Database (Program Manager and Program Administrator)
       - All the database tables will be visible in a tabular format.
         * The PA will be responsible for populating all the tables based on a previously set attributes in the database.
         * The PM will be responsible for organizing and managing the populated tables.
    2. Student Portal
       - This will be the main area where students will be able to sign up for classes
       - The students will be able to see a course catalogue where it will display all the proper course listings
         * Students can select courses and add them to their shopping cart for later review and submission
       - There will also be a area where students can manage their courses in a table
       - Once the student chooses their courses, based on limitations on how many units they can sign up for, they can submit their courses and it will automatically sign them up
  + ***Backend (server)*** 
    - The default server for this application will run on Ubuntu 10.10; a minimum of 512mb RAM and 1.5GHz clockspeed CPU will be assumed.
* MySQL will be used as the database daemon.
* Lighttpd will be used as the server daemon.
* Lighttpd will use FastCGI to interface with the Django framework.

**Performance:**

The scheduling algorithm shall run in no worse than O(nn) time. All other operations in the system, except starting the server, should take negligible amounts of time. Negligible here is defined as an absolute maximum of 1 second. Note that DNS resolution and HTTP transportation is not within the scope of this requirements document; slow interaction with the interface over the internet may be a symptom of a poor connection.

* **Usability –** how the users must be able to interact with the MyCourses system
  + All users will interact with the system through a web browser.
  + They will log in using their username and password.
  + The system should be accessible within and outside of the system’s network.
  + Buttons and links should be simple and easy to read and intuitive for the end user to understand the different aspects of the system.
  + An online help page should be available.
  + Database tables should be active and data should be actively pushing to the end user’s UI.
  + Tables are populated only by the PA.
* **Constraints –** The Course Scheduling System will input the following constraints for   
  courses, classrooms and lecturers. It will then process all of the   
  constraints and find an optimal course schedule for a given school.
  1. **Classrooms**
     + Size of classrooms
     + Classroom can only hold one class at a time
  2. **Lecturer**
     + Can only teach one class at a time
* **Wish list** 
  + We would like to have an application for smartphones that will   
    give the students an interface where they can access the Course   
    Scheduling System. The items on our wish list will make our project more attractive to the vast majority of people using it; the students. Our hopes are to create an interface that will allow the student to use our software as a tool that they can use throughout their college experience.
    - Search for classes

Students will be able to search through the course schedule to browse potential courses.

* + - Add/Drop/Swap classes

Students will be able to modify their class schedule through their phone.

* + - Student Planner

Our student planner will give the students a clear picture of their history, and future at their University. It will give them information on courses that they need to take for their major, and courses that will fulfill GE’s.

* + - Connect to Google calendar

This option would allow the students to integrate their course schedule with Google’s calendar. The student will only have one calendar that they will need to manage, making it much easier to plan events without interfering with their classes.

* + - Connect via Google Maps

This feature will be especially useful for new students, it will pinpoint the student’s classroom within the campus and allow them to see exactly where their classes are on the map. When integrated with GPS it will give them the quickest route to their classroom.

* **Coding Standards –**
* For this project, we'll be borrowing a significant portion of Python's style guide (PEP 0008). PEP 0008 can be found at: http://www.python.org/dev/peps/pep-0008/
* Note that we will be ignoring the section on Version Bookkeeping. Any Python code should conform to this style guide as well as possible.
* We will also be using Django's design philosophy. Django's design philosophy can be found at: http://docs.djangoproject.com/en/dev/misc/design-philosophies/
* Most notable is the DRY principle - don't repeat yourself whenever possible. Additionally, spaghetti code should be avoided as much as possible.

**Preliminary User Interface:**

**\*\*Please note that this was a preliminary user interface when we generated the requirement documents, the interface done in the current version of the system does not reflect the preliminary user interface\*\***

****