Luis Perez & Munkhjardal

Database II

April 26, 2016

Project Specification & Application Techniques

For this project sequence we were tasked with the creation of a relatable database, and implementing functionalities that would be expected of said paradigm. The database was meant to simulate, and deal with the manners in which an administrator could view, edit, and review changes made to said database, and relegating these changes to the database that the queries were being based on. The administrator, and for some stations the students, could access the database through the web interface, and validate their progress so far in the school year. A lot of the functions that would be expected of a system of this sort are easily accessed as well.

For project startup, run the XAMPP Server, and begin with the home file included in the attached zipped folder. It holds images displaying expected output from the applications website side, as importing demeans quality for the word document.

The administrator is able to Insert a student into the database, preincrementing the Student's ID #, so as not to corrupt the database and have multiple students with the main identifier of the SID#. The administrator is also able to check up on a student's progress so as to recommend things that they could do to continue on the path towards graduation, and should the student be able to graduate based on their current progress tell them the great news, if not then it recommends possible steps to rectify the matter.

On the less formal side, we are able to fully grasp the students data, displaying the ones that are able to graduate for ease of access. They are able to check the courses that they have taken so far in their Scholastic career, and all of their basic information oas displayed to the system itself. This is useful, as they can also see an academic transcript that is able to be given to an employer, and do so in a manner that is not invasive, and is also facilitated, it has been a difficult process on the side of the casual use of the current platform that navigation ois cumbersome, and unintuitive.

From the site the administrator is able to enroll a student in courses, and once those courses are taken is able to upgrade the students grades as well. They are, at the time, able to enroll students only in courses that the student is able to take, and are being offered in the semester coming forth, so as not to enroll the student in a class that they do not meet the prerequisites for, or was offered three years past. This delimitation could cause a small issue though, as a student that attempts to enroll midway through the year, due to the limited dataset, may not be able to get into any classes, as they do not meet the needed requirements for them, or they are not being offered at the time they are attempting to get in, to circumvent this, the importing of a transcript for a student is an option, added post doc submission, and allows for a full transcript of courses that are transferred in. This allows for grades from the courses, to if needed, be counted for the student, and allow them to enroll normally. There is also a side point that should a student want to retake a course, as long as they did not attain an A in the Course, then they are free to take it again, and increment their GPA, currently completely overwrites the students data in that course, since we cannot directly mutate the actual dataset, adding in a placeholder to state that it is not the first time they are taking the course would not be plausible.

The main workhorse for the website is more or less the mass of PHP code, that allows direct interactions with the database, and relays the information that is found back to the user. The underlying PHP uses SQL queries to access the database, and populates it when needed as well. All of the base queries that are needed, are readily satisfied, and the ones that aren't are placed in directly are done so through combinations of the PHP code, and the direct user interface of the HTML produced website. There was also a simple template used to relay the data.

There is also a simple phone application that allows the user to access the database directly, and view their progress, as was stated in the specs for it, and do so in a manner that is simple to use, and intuitive. The less actions that the user has to take to achieve their expected outcome the better it is for their experience as a whole. The android application of the Project was created using Android Studio. It connects to the server sending post request with parameter of specific student ID# given by the user reading the response data back in, parse it into Json and display them when the button “SHOW STUDENT INFO” has been clicked.



Two layout, two java classes have been used. First layout is home screen – “main\_activity” where button and field to enter the student ID are. Next layout is main\_activity2 where all the parsed data is shown in small text format.

On start of the application, the class ActivityMain.java is executed and displays the first layout waits for the user to input SID and click on the button. When the user clicks on the button an object of class Display\_info.java is created and send post request to the server with the parameter of SID#. Display\_info.java invokes second display layout and print out all the data on the app. If return is pressed, the app will go back to the initial screen where user can look for another student info.

The student info page displays all the data in the database related to the student.