Trevor Bartlett 010650454 http://csce.uark.edu/~tdb005/FINAL\_DB\_PROJECT/

Lauren Rainbolt 010642413 http://csce.uark.edu/~larainbo/dbfinal/

Database Homework 5

We would like to preface this report by saying that we are quite glad that we began this project early. While it was not the most challenging project content wise, writing up all of the queries, tables, and functions did mean that the work required for completion was quite tedious. We would also like to take the time to mention that all of the pictures used to create the nice background are copyright free and we are intended for fair use through a work that is transformative.

The first task, adding students to the student table, went smoothly. Much of the time spent on this section was put towards making it integrate with the login system. However, this task proved to be much too daunting to perform this early as the login was a much more arduous process to get running then the task at hand. With this under consideration, we decided that we would skip getting the login to be fully functional for now and just construct a skeleton for it so that it could be interacted with from the interface. Once the login had been tabled, the rest of the implementation was easy. We made two input fields so that the user might be able to the ID and name of a student. After this, we created a drop-down box for the student’s major and we propagated it with a few different majors so that we could test its output on Turing. Once we were confident that this was working in the intended manner, we moved on to the next step: adding a course.

After having made the addStudent function, the addCourse function was trivial. This function was largely the same as the last one, but instead of pulling the Student ID and name from the Student table, we were pulling the course name and the department name from course. After we had finished this, we went ahead and implemented the return link that would take the user back to the main page. At this time, it was just a simple text implementation and it would not be until later that we would add it in as a nicer hover button in the header.

The last task that was necessary before moving onto the views was the creation of a function that would allow the user to apply to get into a class. For this function, it was necessary that the user be able to Enter a student ID as well as a course name and then have these values inserted into the enrollment table. Because of the high probability that the user would enter a class incorrectly and the relatively obtuse idea of trying to match strings that users entered against a preset of classes, we decided that the best way to go about having the user select a class would be to just provide them with a drop-down list of all of the available classes so that they could just click on the one that they wanted. Several different ideas came up about how this should be handled. On one hand, it felt incomplete to allow a user to just add as many classes as they wanted to their schedule, so we considered limiting it to five or six. In the end though, we didn’t really have any solid reason to cap it other than the fact that it would feel odd for a student to pick up fifteen classes, so we just left it in.

With the first half of the assignment out of the way, we moved onto the much more conceptually difficult views. To start off, we once again attempted to get the login to work with each of the views. This once again did not pan out and it was pushed back to the end of the project. After a refresher course on W3 schools about how to make tables look nice, we were ready to begin. As far as getting the actual information displayed, this task was very easy. Instead of inserting into the database as we did in task one, we were pulling from the database for each student. Aside from one small incident where the program was attempting to pull from “Students” instead of “Student”, this task went very well.

The next task was much the same, but this time we had to pull the Course ID, Course Name, and Department Name from the Course table. This task was effectively the same setup as the previous one, but at some point, during the view tasks we did take a break to reformat the database tables that we had previously established as they did not fully cover everything we needed. We also added a nice copyright free image to our project and made sure that our buttons were centered on the homepage so that the overall presentation would be nice. With this completed, we moved onto the final function.

The final task was the largest of the three views. This task required that we pull from both the Course and Enrollment tables to get the course id, the course name, and the department name. The student ids in the course and enrollment tables also had to match for the correct data to be displayed. When we created the StudentId variable earlier in the database, we limited its size to eleven characters so that garbage infinite-number strings could not be entered into the system.

With all the tasks completed, we still had several hours left until the deadline. We decided to use this time to make our main page better. We cleaned up the overall layout and made all of the options use hover buttons. We also decided to incorporate CSS into our code. We decided not to use a CSS style sheet as we knew that we would be making several revisions to the style. In order to do this, we ended up doing three divs. The first div that we used was the top-nav div. We used this on every page to be able to move back to the main page. The next div that we used was the class = center. This took all of the menus we had and centered them horizontally on the page. Using the border radius attribute of CSS, we w3ere able to make a background for the menu. The final div we used was the class = button. The button CSS created oval shaped buttons that had a hover component that made them feel a lot more interactive.















