**Part I - Project Information**

**Introduction**

Our database system project “Database Academy” was designed to be an internal search tool for a high school (inspired by the UVA Internal People Search website). In order to create an effective search tool for a high school, we had to accurately model several entities and their relationships. The entities themselves consisted of two types, people and activities. The three people types that we attempted to model in the system was the students, teachers, and parents of the students. In our system, each parent is associated with only one student, while students may be associated with one or many parents. Students are indirectly associated with teachers through the courses they take. Our activity entity types consisted of courses and clubs. Exactly one professor taught every course in the school, but every course could be attended by multiple students. Additionally, each student could belong to many different clubs, but each club is sponsored by exactly one teacher.

The internal search tool also had to be safe and protect the private information of the users.

Introduction (2%)

- Describe the world you are trying to model. Include everything that you feel is relevant. This should be around 1/2 to a full page.

**Requirements**

* If the user is a student attending Database Academy, a teacher teaching at the academy, or a parent of a student at the academy, they should be able to log in using their email and a password (we assume people logging in already know a username/password combination for a student/teacher/parent they are trying to log in as).
* Users should be able to log out.
* Any user should be able to log in and go to the home screen of the website, which features a menu bar (which is consistent across all screens of the website), and a search bar.
* From the home screen, any user should be able to search for a specific query in the search bar and be presented with a page that lists all of the names of people (students/teachers/parents) whose first or last name match the search query.
* Any user should be able to click on any one of these names and be presented with details of that person, which is unique to their role at the academy.
* \*\*\*Students should only be able to view their own GPAs.
* \*\*\*\*Parents should only be able to view the GPAs of the students associated with them.
* \*\*\*Only teachers should be able to view their own salaries.
* \*\*\*Only the currently logged in user should be able to see their own password and email.
* Any user should be able to navigate to the search page for Clubs and search for specific clubs at the academy. Once they are presented with the list of clubs, they should be able to click on any club name to view the specific details (this same functionality should also exist for courses).
* Teachers should be able to update, delete, and add new information to the website for any type of person, club, and course (except salaries of other teachers, which they cannot see).
* \*\*\*All of the data in the database should be able to be exported by a Teacher as a \_\_\_\_ and printed. The salaries of other professors should not show up.
* \*\*\*\*Special Functionality

**Part II - The Design Process**

• Explanation of design decisions that you made in creating your application (i.e. why you choose the type of app you did, why that language, if your database should be secure, did you secure it and how, anything interesting or relevant, etc.) (2%)

- Describe your thinking as to why you did certain things (around 1/2 a page)

• An E-R diagram (5%)

- Make sure you model EVERYTHING in your database! Don't leave out stuff!

• Database schema including integrity constraints (2%)

- Just give the schemas straight from your database with things like primary key, unique, not null, etc. Easy 2% here.

• Proof that your database is in Third Normal Form (5%)

- Take your tables and perform the tests we went over in class. Be thorough. Do NOT skip steps.

**Part III - Evaluation of Product**

We first tested the login functionality of our program. First, we tried logging into the website with the correct username/password combinations of various students, teachers, and professors to make sure that all user types had access to the website. Then, we tried logging in with correct usernames but wrong passwords, correct passwords but wrong usernames, and other incorrect username/password combinations to make sure that only the correct users had access to the system.

Next, we tested the search functionality for our system. For each user type (student/teacher/parent) we navigated to the home page and tried different searches. We matched the results of the search to the database to see if the list of people on the results page was actually the people that should have shown up on that search. We also tried searches in which we knew no data from the database would match to and made sure that the correct empty result page was shown. We repeated this process for the courses search page and the clubs search page.

Next, we tested the privacy/security aspects of our system. First, we logged in as a student and made sure we could only view our own GPAs and no other students. We also made sure we couldn’t see the salaries of any of the professors, or edit/update/delete any information from the website. Next, we logged in as a parent and made sure we could only see the GPA of our kids and none of the teachers salaries. We also made sure there was no way for us to update/delete/edit any information. Lastly, we logged in as a teacher and made sure we could not view the salaries of any teacher profile besides our own. We also made sure we could edit/update/add information for any person, club, and course (except the salary info for other professors).

Lastly, we made sure we could easily navigate from Home (People Search Page) 🡪 Course Search Page 🡪 Club Search Page. Then, we tried logging out as a user to make sure it was possible and actually logged us off the website (didn’t allow us to view anything anymore).

Sample Data and Queries:

• Sample data and sample queries from your application (2%)

- Give an idea of what your database does without having to load it.