1. **Executive Summary**
   1. **Purpose of this document**

This document will outline the design processes that took place while developing a Django application to interface with an existing database. This project will satisfy the “DB-Backed Web Application” requirement for the course EE468.

* 1. **Identification**

Matthew Lukaszewski, Josh Ingerowski, Stephen Miner, and Lucas Barnello all contributed to the contents of this project.

* 1. **Relationship to Other Plans**

The Django interface interacts for simple queries based on the user's input of username and password. The options were restricting the database users themselves to specific queries or interfacing the Django application to restrict what each user can visualize. The latter option was chosen to simplify the development process.

* 1. **Methodology, Tools, and Techniques**

To develop this application, pre-established SQL databases were leveraged. Django web-interface implementation was utilized to provide a pleasant user experience.

1. **Design Overview**
   1. **Background Information**

Pre-established SQL databases were leveraged to allow users to view the contents of specific databases through a Django-Web application. The database utilized is designed around a university structure and includes tables and data on the following categories:

* Student
* Instructor
* Prereq
* Section
* Takes
* Teaches
* Department

For the purposes of this project, we only used the tables student, instructor, teachers, department.

* 1. **Current Process**

As of right now if a student was to look at the database tables, they could query and modify all of the tables in the university database.

* 1. **Proposed Process**

Using Django, the proposed process would be to have the user login. Once the username and password is entered correctly, a page respective to that specific user is loaded. When logging in the user will have five attempts before getting a timeout on their machine.

The respective user web pages are restricted to specific lists generated from the databases. Specific list views can then be modified based on the user input.

* 1. **Constraints**

There are only three registered users; student, instructor, administrator. Each user will be constrained to their respective pages, which are constrained to show specific contents for each user.

Some of the user imputed data may be constrained depending on the data in the database. If a user is to input a variable value that is not found in the database an error will be raised to the user's attention.

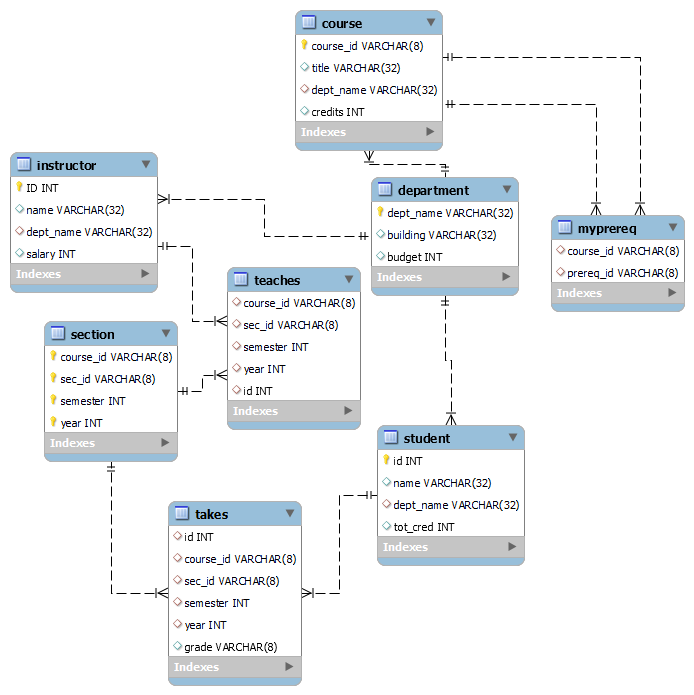
* 1. **Design Trade-offs**

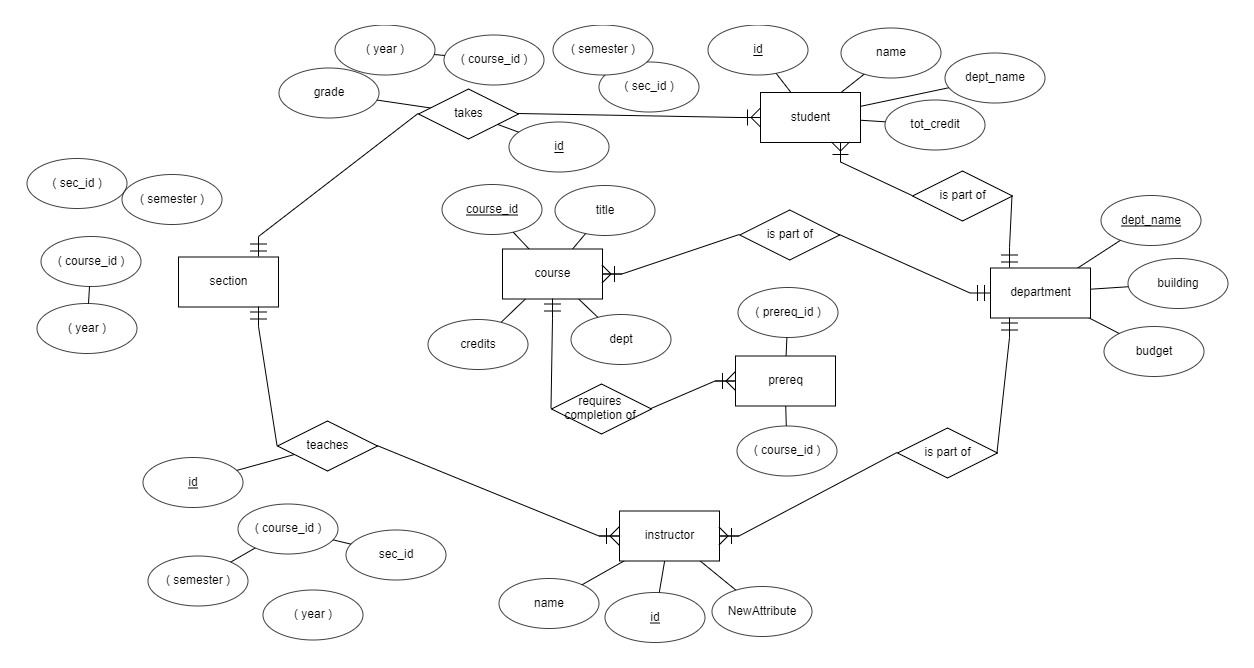
Constraining the users to their respective data through the database was considered. Ultimately it was decided that restricting the users through the database would make further implementation and improvements difficult. Instead the users can only modify which data they view through the web application.

* 1. **Security Assurance**

Security Assurance was a problem of great concern when developing this software, as unauthorized access to any user’s account may potentially leak sensitive data stored within the database. To mitigate risks of unauthorized access we have implemented a login system that will automatically check user credentials and give them database permissions accordingly. As an added security measure, after ~5 consecutive failed login attempts the user attempting to sign in is temporarily locked out of the program (this may not be working at the time of submission). This extra measure attempts to nullify brute-forcing attempts and keep user account info safe. CSRF tokens w/ random generation and validation were also implemented to prevent CSRF attacks.

1. **System Architecture**
   1. **Database Schema and ER Diagram**

****

****

1. **External Interface Design** 
   1. **Interface Architecture** 
      1. Login Page
         1. Admin Page
            1. A table will show all professors and can be sorted by name, department, and salary.
            2. A table will show the min, max, and average salaries by department
            3. A table will show the professors names, department, and total numbers of students taught by the professor in a given semester.
         2. Student Page
            1. List the course sections offered by departments in a given year and semester.
         3. Professor Page
            1. Enter specific Professors name
            2. A list of course sections and the number of students enrolled in each section that the inputted professor taught in a given semester
            3. A list of students enrolled in a course section taught by the professor in a given semester
   2. **Interface Detailed Design**

A login view will be generated to prompt the user to login in with a specified username and password.

Each user, after logging in successfully, will be brought to their specific viewable change where they can view their respective tables. Within each user page, inputs will be prompted to change the view of the tables. There will be restrictions on which parts of the university database the user will be able to view.

The admin page will consist of three tables:

* The first table will show all of the professors, sorted by name, department, and salary. The admin will be able to change the table by choosing how they would like to sort the table.
* The second table will not have any user input and will list the minimum, maximum and average of salaries, sorted by department.
* The third table will show the professors names, department, and total numbers of students taught by the professor in a given semester. The admin will be prompted to select a semester to change the view of the table.

The professor page will consist of:

* The professor will be prompted to enter a professor's name.
* A list of course sections and the number of students enrolled in each section that the inputted professor taught in a given semester. The professor will be prompted to select a radio button to view the Spring and Fall semesters data.
* A list of students enrolled in a course section taught by the professor in a given semester. The professor will be prompted to select a radio button to view the Spring and Fall Semesters data.

The student page will consist of one table:

* List the course sections offered by departments in a given year and semester. The student will be prompted to enter a year and select the Spring or Fall semester.

If there is an input by any of the users that does not exist in the database, an error will be raised to the users attention.

1. **Human-Machine Interface**
   1. **Design Rules**

When the application is first run, the user will be prompted to enter a username and password. As of right now, the three users; student, professors, and admin have been established with their own set password.

When the Admin logs in, they will be brought to a web page with three different tables. Each table will have their own purpose and some user input may be required to show specific data in the table.

When the Student logs in, they will be shown one table that lists the course sections and

* 1. **Inputs**

Each user will have a specific amount of inputs they are required to input to view specific contents of each viewable table. Students will be able to input \*\* to show outputs for their table. Professors and Admins also will have the ability to input \*\* to show outputs for their respected tables

* 1. **Outputs**

Automatically the tables for each of the users are filled with default contents. Given that there is user input, the respected tables may change based on the users input. For the purposes of this project, the table's contents are restricted respectively to each user.

1. **System Integrity Controls**

**\*\* if we need appendix we should add one**