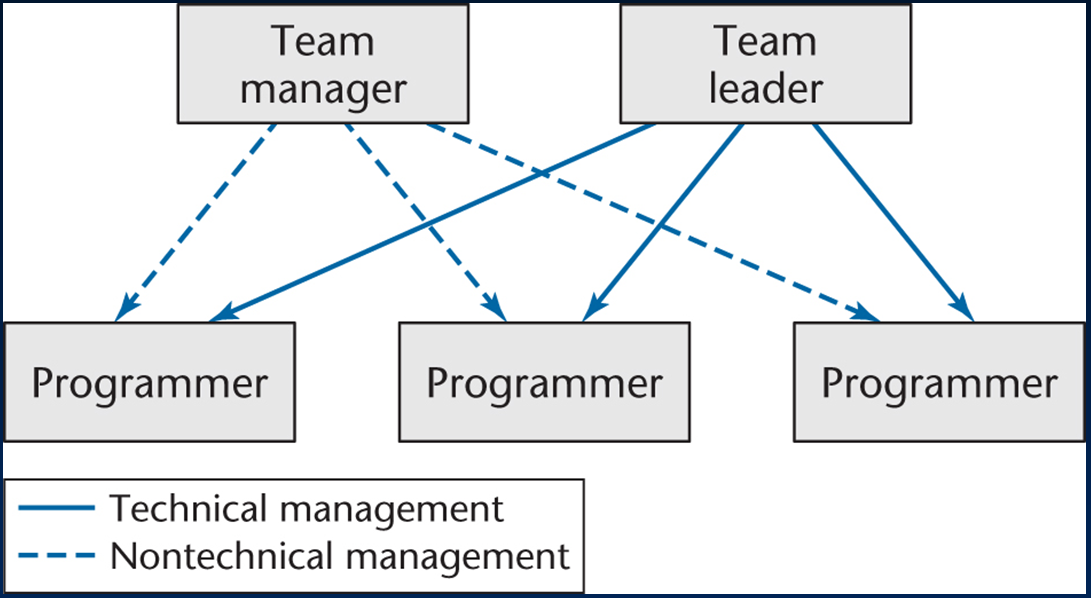
# Software Engineering Report

## Group Structure

Our group started with a democratic group structure and evolved into a chief programmer structure with Carlos Rios doing a lot of the base coding and Ivan Sosa handling some of the non-technical responsibilities. Our Group settled on the Democratic team model

#### Democratic Team

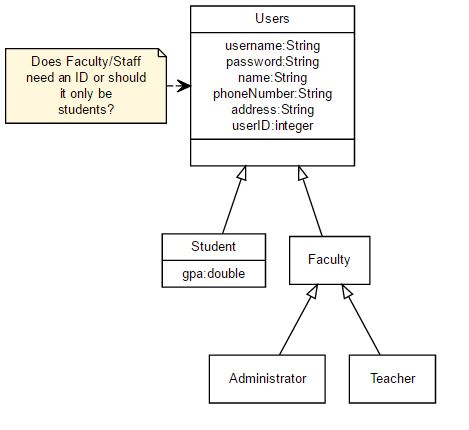


* Team Leader Carlos Rios
* Team Manager Ivan Sosa
* Programmer Satish Subramanian
* Programmer Phillipe Hehn
* Programmer Derek Jones

## Diagrams

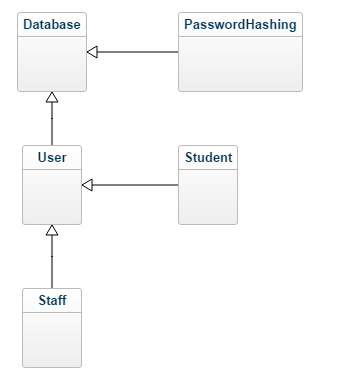
### Class Diagrams

#### Class Diagram – Version 1



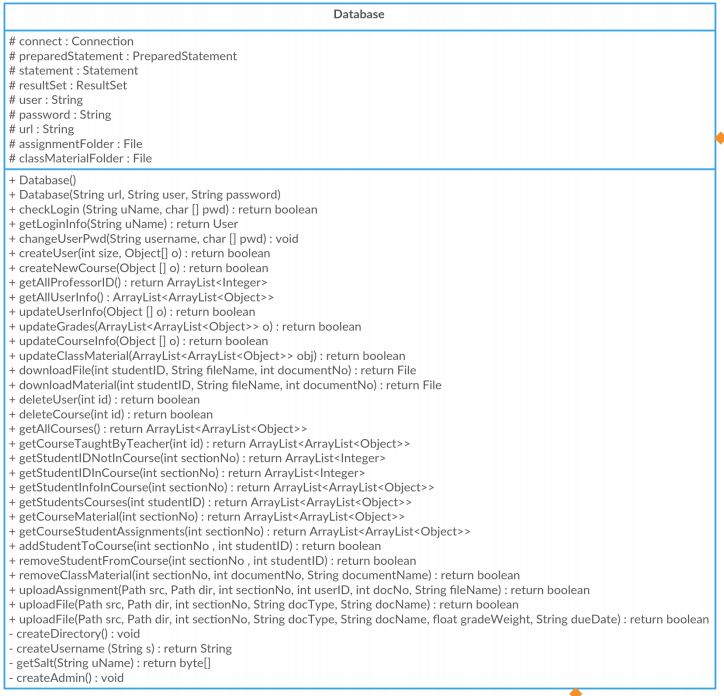
Our first diagram was the class diagram. Here we outlined the users for the program and what keys needed to be put for each user. These users would be used later in our ER diagram.

#### Class Diagram – Version 2

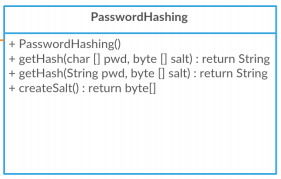


The final version of the class diagram had the final classes and where they are derived from. On the next pages are an in depth look at each class and what goes into each one.

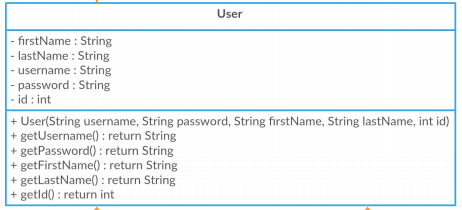
**Database**



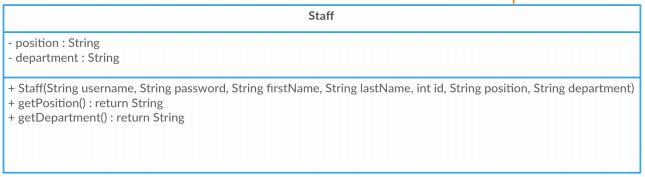
**PasswordHashing**



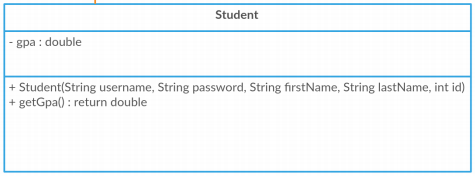
**User**



**Staff**

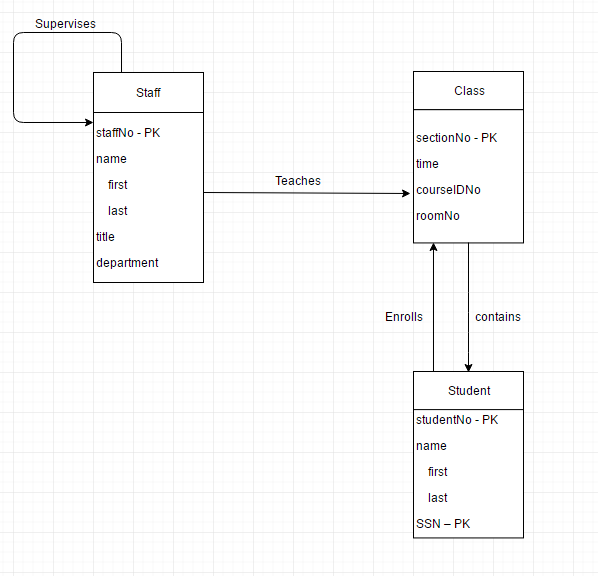


**Student**



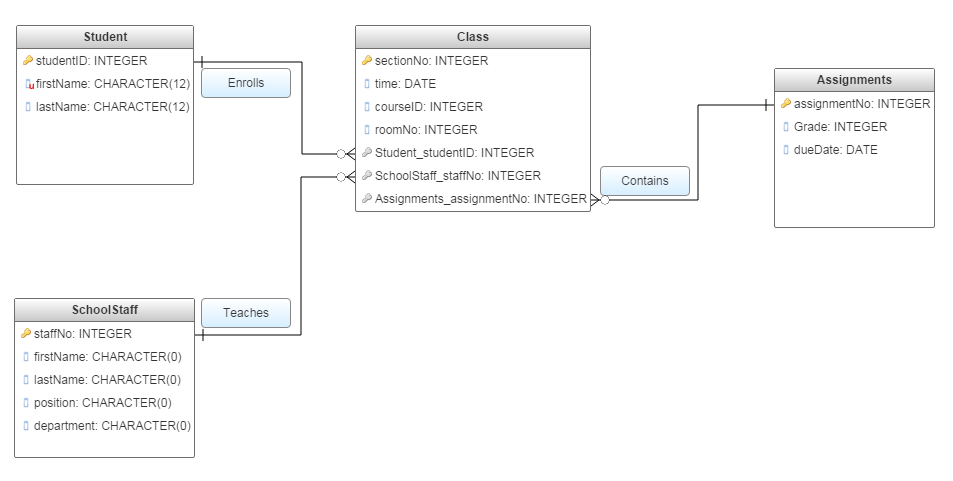
### Entity – Relationship Diagrams

#### ER Diagram – Version 1



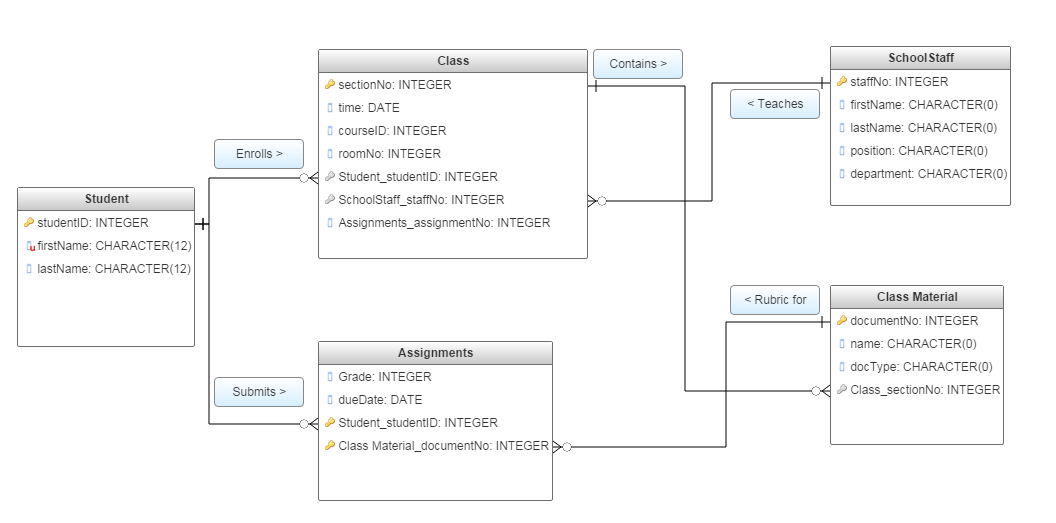
Our first version of the ER Diagram had a basic outline for how our entities would interact. There were a lot of errors and missing entities.

#### ER Diagram – Version 2



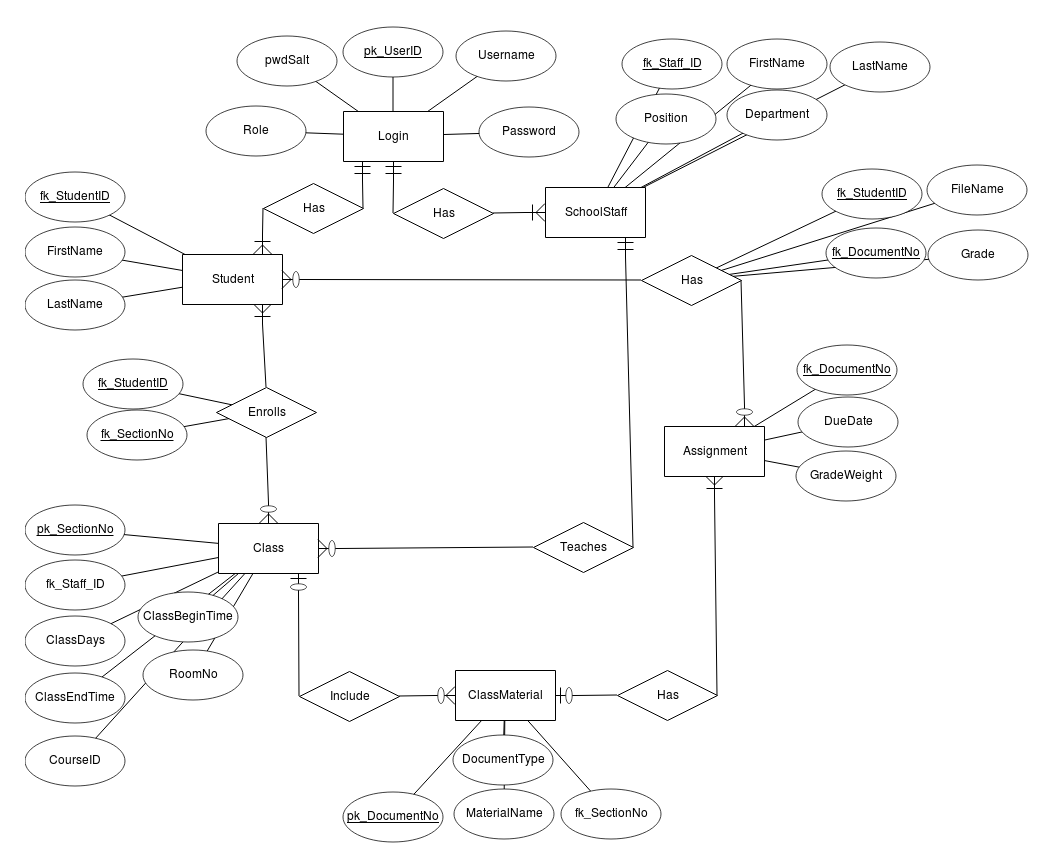
Our second attempt had a more clearly defined relationship between entities. We still lacked a way to relate assignments back to students to turn in. We decided to introduce another entity to have everything fit together better.

#### ER Diagram – Version 3



Our 3rd diagram had a way to handle a unique assignment turn-in for the class materials. We didn't have a fully realized scheme yet so we made further adjustments in our 4th version

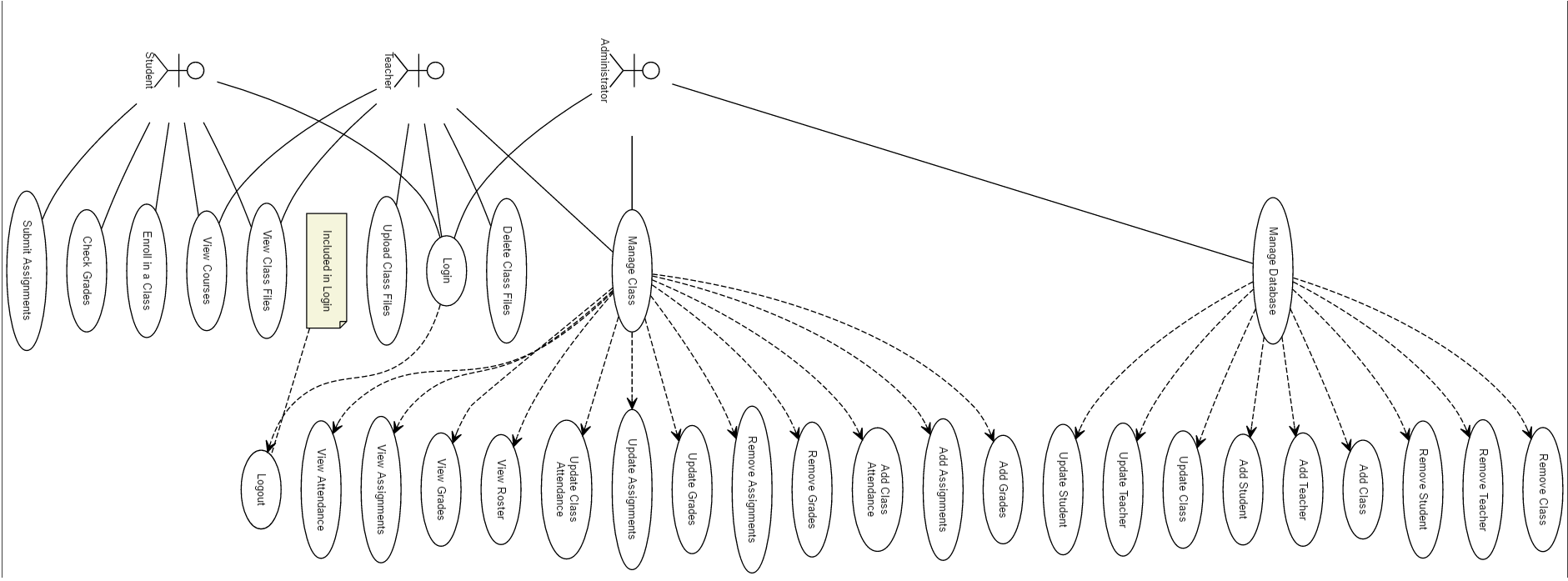
#### ER Diagram – Version 4



The final version of the ER diagram included a way to handle a login. We ended with 6 entities: Login, Student, SchoolStaff, Assignment, Class and ClassMaterial. This diagram had everything we needed for our SQL schema and eliminated redundancies.

U**se Case Diagram**

Our use case diagram contained all actions that could be taken by users. The 3 main users were the Admin, Student, and Teacher.

Teacher needed to be able to add, modify, and remove assignments and grades. They also need to be able to view the class roster, as well as upload files and delete files.

Student uses cases were to view reports, course assignments, grades, and uploaded files. Students also needed to sign up for classes and submit assignments.

Admins needed all the privileges that Student and Teacher had, but also needed the ability to add, remove, and modify Classes, Students, and Teachers.

## Conclusion

Though our group easily went through the requirements portion of the development process, we encountered issues during the analysis portion. Our project grew and evolved from what we initially planned. Our diagrams had to be modified to include functionality for all the use cases we outlined during the requirements stage. Even though we spent more time than anticipated on analysis, we progressed through implementation quickly since we had most of the details of our project planned out.