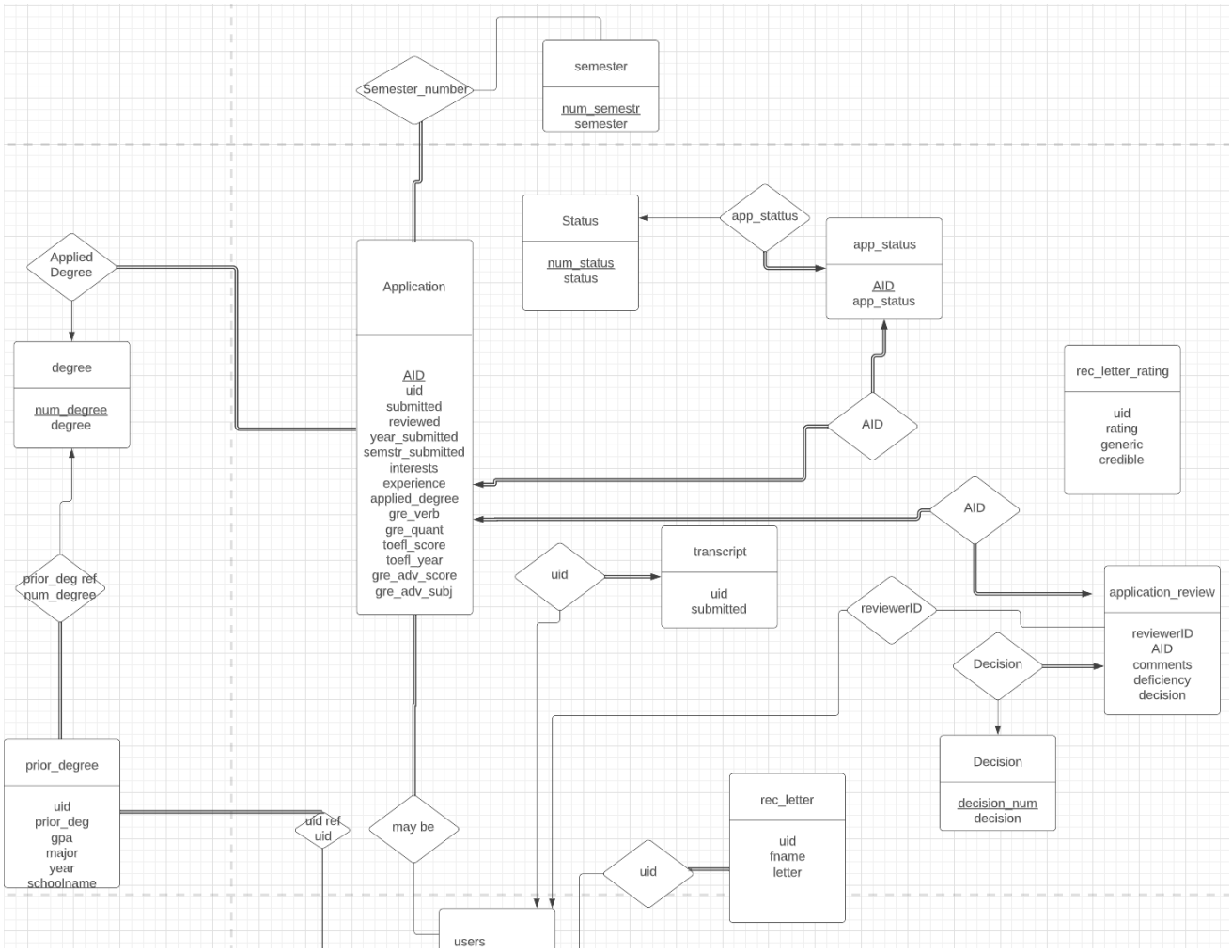
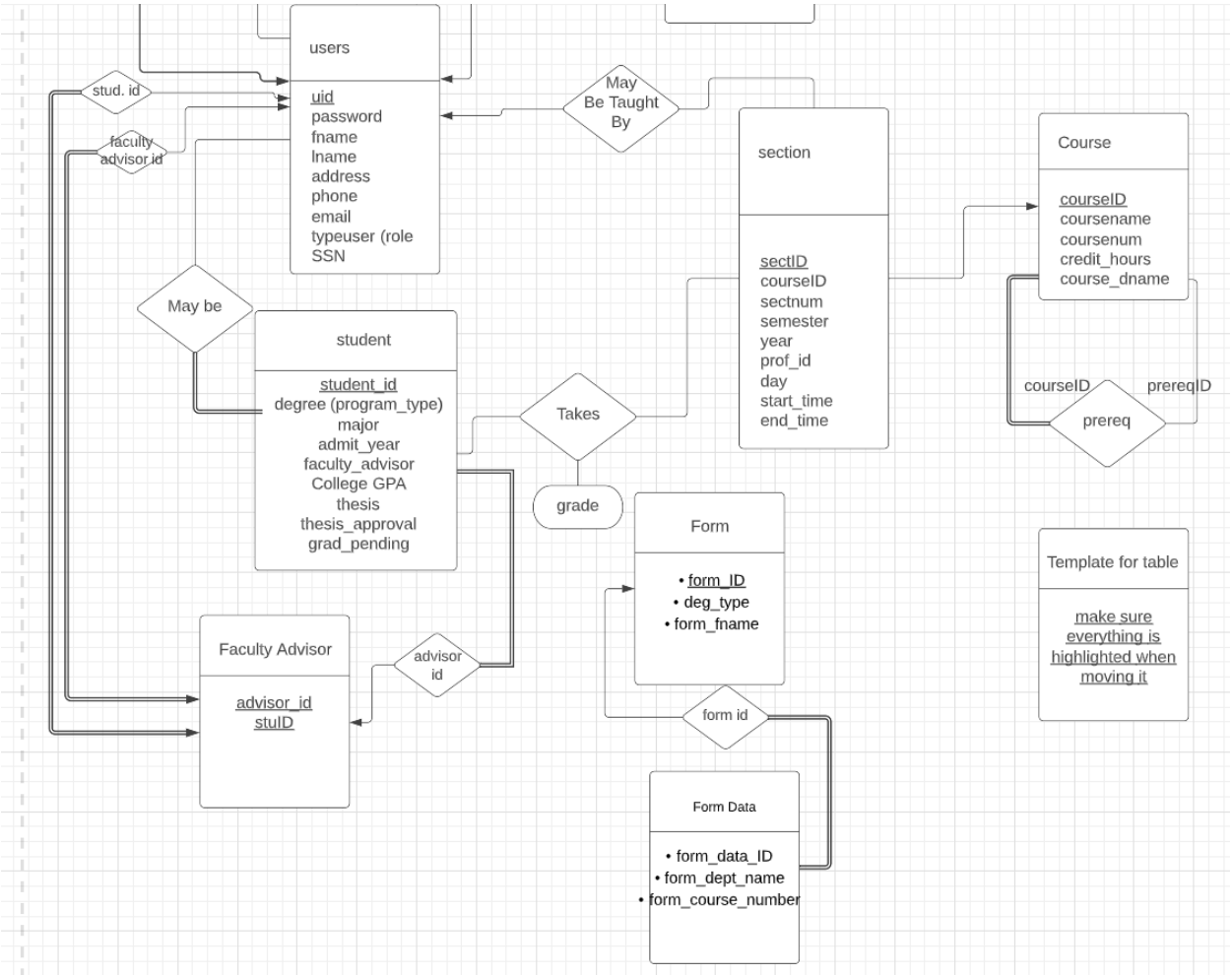
Phase 2 Project Report

Tanim Khan, Jay Samaraweera, Karl Simon

ER Diagram: Tanim, Karl, Jay

https://lucid.app/lucidchart/fd3a1eb2-1ac3-4511-b9f1-fa3d58c69aa5/edit?invitationId=inv_58ef8544-33d0-4ce9-84aa-e66a5407df1f





Normal Forms of Schema: Tanim

The tables originally from the REGS system, users, student, section, course, prereq, and takes, were able to reach 1NF since attributes were atomic, 2NF since the entire primary key was needed to access a row, 3NF due to a lack of transitive dependencies, and all of the tables, except for the prerequisites table, reached BCNF since there were no dependencies with the right hand side being part of a candidate key.

Most of the tables originating from the ADS system, form, form data, and faculty advisor with both attributes as its one primary key, were able to reach 1NF, 2NF, 3NF, and BCNF for the same reasons as in the previous paragraph. The items in form data all form one primary key in the SQL code even though it was not displayed in the diagram.

The remaining tables come from the APPS system. Some of them reach BCNF due to the same reasons in the first paragraph, but a lot of them do not have any primary key of their own, meaning duplicate rows can exist. While this is not desirable for most tables and a major oversight on our part, the rec_review_rating table should be able to have duplicate rows since multiple people can give the same ratings to the same applicant. Fortunately, most of these

keyless tables would not have duplicate entries in a normal run through. That being said, a fix for the lack of primary keys would be to make 1 or more attributes a key and adding logic to account for said key in the python code.

Visual Overview
Karl: https://vimeo.com/707146009
Jay: https://vimeo.com/707183051/f7f53310db
Tanim: https://vimeo.com/707183219/2ffb9e8a33

Design Justification: Jay, Karl, Tanim
Application Status: It made sense to chronologically follow a timeline of a student submitting an application to then being able to view their application status. Having the option to view an application status when an application hadn't been submitted yet made no sense to us. Thus it would show up only after an applicant had submitted their application.

Unique SSN: Makes sense for everyone to have a unique SSN. Making the SSN a unique column in the users table also helped with functionality for the forgot password function (worked on by Karl) so that the password is retrieved and changeable when giving information only the user could verify.

Users: the database schema relied on all users coming from the same user table. This allowed for much easier creation of new user types, and simplified fetching of data for users. For the merging of ADS and APPS, it was much easier to have a single user table because the same user (i.e., same database entry) could progress from new applicant to graduate without having to move them across tables.

Dashboard: the difficulty with merging ADS and APPS was to coordinate functionality and display it seamlessly together on the website. The solution was a generic dashboard for each user. Once logged in, the user would be presented with many links/buttons that take them to the selected feature. This meant that adding/merging ADS and APPS functionality was as simple as adding a new link on the dashboard, thus making merging easier and allowing the ADS and APPS integration to occur at their own pace.

Main structure: For the site, we decided to base it around the REGS website from phase 1 because the databases for it and the ADS systems were heavily intertwined with features implemented in REGS. It also made for a good foundation to build the layout and CSS on.

List of Design Assumptions: Jay, Karl, Tanim
Rec Letters are requested from the university after applicant has an application** This was assumed because logically it wouldn't make sense for a student with no application recorded in the university to send recommendation letters. In chronological order the student should make an application to the university so that they're tied to the application, then request recommendation letters from their respective recommenders.

3 Recommendation Letters required before being eligible for review. Application is only considered complete when:

a. Transcript is considered submitted by GS
b. The applicant has at least 3 recommendation letters

3 reviews for application before it is eligible for final decision The GS could only see complete applications that have been reviewed 3 times by 3 different faculty reviewers. The GS could see the decisions from the other reviewers on the review form and administer a final decision based on that.

If accepted, an applicant was now a student and could register for classes If an applicant was accepted, it was now a student, which meant that they were now redirected to a student/REGS portion of the website when logging in

Form1 functionality: to submit a Form1, a student must have registered for (IP or graded) all the courses they list on the Form1. As a result, the Form1 is a formal way of reiterating the courses a student took (or are taking). Then, once applying to graduate, the audit checks if all courses listed were completed, as well as grades, etc.

Section ID vs Course ID: the section ID for a course is the id that refers to the course and the semester it's offered in. The CourseID is unique to the course only. So for a particular course, there are 4 Section IDs, but only one Course ID.

Section ID vs Section Number: Section ID is a unique integer that represents each section in the database. Section number is there to help distinguish a section from other sections with the same course id.

It was assumed that there would be multiple semesters in the system, so SQL statements were added to insert several more sections. To prevent time and prerequisite conflicts between different semesters, the REGS code for such errors had to be expanded on. You cannot register for two sections of the same class simultaneously or retake a course. However, if a student fails in a course, they can retake the course in a later semester.

We also assumed that since there were instructions to put people such as Narahari as a faculty reviewer, instructor, and advisor, that there should be three users for him with different types to each since the design was for each user to have exactly one set of permissions, meaning advisors were not expected to teach courses or review applications. There were some students who had the same name, like the Ringo Starr for the APPS and ADS system each, and multiple students with the ID 66666666. One student was assigned 66666667 since IDs had to be unique.

Special Features Karl, Tanim, Jay

1. Update Advisor for Student: each student is assigned an advisor upon applying, but the GS has the ability to change faculty advisors for students. Under the Students link on the

GS dashboard, the advisor ID can be changed for each student by typing in a new faculty advisor (typing in 0 will remove the student's faculty advisor).

2. Forgot Password/Password Reset: when logging in, each user can select "Forgot Password" to reset their password. They will need to enter the unique SSN, fname, and lname, and then they will see their old password and a text field to add a new password.
3. Semester display: for the pages displaying all sections. An input form was added to the top, requesting a number to be inserted. Once a number ranging from 0 to 3 is added. It displays all the sections taking place during said semester. For instance, if I put 0 in, I would see all the sections taking place in Spring 2022, which is the first semester of our database. A similar feature was added to the course registration page to reduce the clutter of information.
4. Partial session checking: Some pages, mostly from REGS, had if statements that made it so that you cannot enter pages without being logged in, and other statements from this portion were added to prevent a database anomaly from occurring. This anomaly was the act of adding a student to the user table but not the student table. The former was partially implemented in other pages from ADS and APPS, but the latter was largely excluded.
5. Add by section ID: A form was added at the very top of the course registration page that enables you to add sections by their section ID. Clicking on the button will add that section to the basket; it works just like the add to basket buttons that were already present.
6. Drop all in progress classes: Link was added to enable you to drop any class you registered in, provided that you haven't been graded on it.
7. Uniquely generated UID: UID was uniquely generated via a function using randint and specifically run through a while loop with a boolean to ensure that no two UID were the same. The UID was later used as the username for a respective account to log in with, as well as an overall cursor for identifying applicants, applications, etc.
8. Saving information in the apply/edit application: After submitting an application, applicants can go back to their application to edit the saved information that has already been inputted, or put new values in fields that have nothing entered in them (i.e prior degree)

Work Breakdown Jay, Karl, Tanim

- *Everyone helped to add to the report

Tanim Khan: Worked on CSS and made a page for the alumni. Provided the groundwork for the final site and helped with integrating both APPS and REGS. Helped with catching and fixing bugs throughout the site. Added a lot of code for populating the sql. Set up the database on AWS, made the repository, and the trello board. Added search functionality to a page displaying the status of students and on another that showed the courses each student took. Added a way to make the display of course sections easier and a page to display all of the users currently in the database as a system admin.

Jay Samaraweera: integrated APPS with REGS in which merging of the login/register, grad secretary privileges, assignment of student from advisor, etc was required. Unique UID function was integrated as well, with form checking for the register and redirection for different dashboards depending on the session[‘typeuser’] variable in the login function. Worked together with the team to have a fully working merged schema (as expected) Also had to account for a lot of missing functionality in phase 1, such as the entire reviewer/GS/CAC portion and major bugs.

Karl Simon: integrated ADS with REGS, which involved adjusting where/how the ADS functionality gets its data from. In addition to merging ADS functionality, such as form1/graduation process, I rebuilt the Grad Secretary page, added the Faculty Advisor webpage and functionality, and improved the sysadmin page. Helped with website styling consistency, and added extra features 1 and 2. Added js form validation across the website.