Part 1 (System Design) :

System Administrator:
➔ Has an account
➔ Has personal information (◆ Unique ID ◆ First name ◆ Last Name, ◆ Address)
➔ can create new users of any type
➔MAY NOT CHANGE GRADES
➔ Has access to accounts of Grad Secretary, Faculty Instructors, Graduate Students

Grad Secretary:
➔ Has account (system provides login)
➔ Has personal information
➔ can view current student's data (probably not past students is those are stored)
Note: MUST DIFFERENTIATE BETWEEN STUDENTS AS PAST OR PRESENT
➔ cannot create new users.
➔ Can modify grades entered by Faculty Instructors.

Faculty Instructors:
➔ Has account (system provides login)
➔ Has personal information
➔ enter grades for students only in the courses they teach, only enterable once.

Graduate Students:
➔ Has personal information
➔ view only their information, add/drop classes

Part 2 (Web interface) :
Pages viewable by user state (logged in/out) and permission level:

Logged out user (not logged in):

- Login page

Logged in user:

Systems Administrator:

- Main page
- Courses page
- This Admin's Personal info page
- "All other accounts" page

(note about the "all other accounts page***

*****accounts listed by permission type -- simulate logging in as another user via a session variable for permission level and a session variable for "view as". When any person logs in, they

are assigned their permission level as they are stored in the database, and their “view as” session variable matches who they are. When an admin or grad secretary selects another user’s page, they modify their “view as” session variable so that they see the data of someone else, but we can also modify what they can do still on a given page.

Example:

A student bob with uid 1234 can log in and view their grades. Bob’s “permission level” is student and “view as” is his user id 1234. Bob as a student cannot change his grades.

A grad secretary john with uid 5678 can log in. This sets up the “permission level” as grad secretary, and “view as” as 5678. As a grad secretary, John can edit Bob’s grades, which bob cannot do himself.

However, this means that John does need to be able to view Bob’s grades in a similar fashion to how Bob views them, just with write access enabled. So, when John selects Bob as the account to view, we set John’s “view as” to 1234, which allows him to see the pages Bob would see, but because John’s permission level has not changed (still listed as grad secretary) John can modify Bob’s grades.

***end note)

- Another person’s personal information page
- A student’s My Grades page
- A faculty instructor’s My Courses page (courses I tech)
- Create an account page
- Home (goes back to view as a Systems Admin instead of any other user type -- only visible when view is as a different user type)

Grad Secretary:

- Main page
- This Secretary’s Personal info page
- Current Student Accounts (derived from “All other accounts” -- but only current students, and no other accounts)
- Another person’s personal information page (only from current students)
- A student’s My Grades page
- Home (goes back to view as a grad secretary instead of student -- only visible when view is as a student)

Faculty Instructor:

- Main page
- This Instructor’s Personal info page

- My Courses page (courses I teach -- should list the students enrolled in the class and their final grade. Note, current classes receive grade “IP” for in progress)

Graduate Student:

- Main page
- This Student’s Personal info page
- This student’s My Grades page
- Courses page (where searching for, adding and dropping occurs)

Total Unique Pages:

******each page may need to confirm that a session login variable (uid) is set, and that if it is not, to redirect to the login page. As well, we may need to confirm that a session userid matches a uid in the database. This will prevent a lot of possible website breaking by messing with session variables. We cannot just trust that if a session variable is set that the user logged in, nor can we simply trust that a user will only navigate to a page of ours by using the links we provided.******

Main page

- Query the permission level of the user and use the user id as the indicator to determine which links can be displayed on the nav bar.

Generic account personal info page

- May need to see if the view as and permission level match. If they do, then the personal info page should be editable. Otherwise, it may or may not be editable depending on whether or not the supervisor (grad secretary and systems admin) accounts have write access.
- If editing, we need form validation on all postable aspects. This can be simplified with dropdown menus to reduce free input fields.

Create an account page (restricted to system admins.)

- Need to query the db account tables to determine if an account exists. If it does, we cannot create a duplicate account for it. Otherwise, the account is likely valid.

Other accounts page (appearance varies by user account level)

- Need to query the database for the list of accounts, and sort them by permission levels.

My Grades

- Query the database for the given user, this link will only be visible to someone viewing as a student.
- Query the database for the given user, if the permission level is a student, grades are not editable. Instructors do not have access to this page, and instead modify grades though the “my courses”

page. A grad secretary will have access to this page, but with the ability to modify the grades. This should most definitely be done by dropdown menu to avoid free form input issues.

My Courses

- Query the database for all currently and previously enrolled classes and their grades. Include a search bar (form validation required) to allow for searching of classes, maybe a dropdown to filter to specific departments.
- Can only insert a course if there is a 30 minute gap between the start and end time for the new course, and any other course whether already registered for or not.
- Must confirm that corequisites are selected to register for, that a lab section is selected if required, and that any prerequisites are satisfied.
- Must confirm that graduate students can only register for classes at the 6000 level.

Home (not shown for user accounts that can only view their own account)

- No checks required. This is really just a link to bring a supervising user back to their index page as opposed to the supervisee's page. -- In other words, modify the "view as" session.

Nav bar:

The pages visible to each account type are listed above. The one difference is that any page listed as "another person/student's x page" is not going to be present on the nav bar, and will only be visible once selecting a user of that type to view. For example, as an admin, you will only see the following pages:

- Main page
- Courses page (full list of courses)
- This Admin's Personal info page
- Create an account page
- "All other accounts" page

The additional pages shown below are only visible when the admin selects the "all other accounts" page, and selects a user of a different type. Then, the links appropriate to that user will appear.

- Another person's personal information page
- A student's My Grades page (the grades they have for the courses enrolled)
- A faculty instructor's My Courses page (courses I teach)

- Home (go back to view as a system admin)

Part 3 (Division of labor) :

Items to be completed by each person are highlighted in a specific color:

Ada: Highlighted in Pink

Cassell: Highlighted in Green

Sam: Highlighted in Blue

To Do List

- Create transcripts page -- break down of the previous classes taken, the grade received, the credit for the course, the total hours of classes take, the GPA etc
- Create sign in page
- Create index page
- Create personal info page (note, the personal info page requires significant form field validation-- dynamic page generation will make this easier.)
- Create the “all other accounts” page
- Write the navbar php (all links and signout) (should be dynamically generated) -- using “require_once() in all pages”
- Generate styling for buttons
- Generate styling for text
- Generate styling for page layout -- whoever does this, let me know what element you want used for each nav bar item, that way you can style it as you’d like.
- Allow system admins to create accounts (we can either have it be that the creation of an account does not require inserting any personal data, and that when a user logs in they can modify their personal info)
- Confirm that a user navigating to a page has the appropriate permission level to see that page (only needs to be written in one php file and then “require_once()’d for all the pages)
- Generate tables based on our draft schema
- Insert data into the sql tables once generated. (Modify the sql code so that when sourced, that data is added by default -- includes “drop table if exists, insert into tableX lots of values”
- Create the “my courses” page for instructors to view the classes they teach -- when an instructor modifies a grade, it should set the boolean typed

"grademodified" variable in the enrollment table to true, which should then disable further modification of the grade for that student

- Create a "register for classes page that allows users to view the full list of classes that exist, but enforces registration requirements: (30 minute space between each class start and end, 6000 level and above for grad students, satisfies prereq, coreq and lab rules, and the student capacity is not at max -- lots of sql queries)

Part 4 (Question list) :

1. **Both grad secretaries and systems administrators list "access" to some or all data. Is "access" read only permissions? Or is it both read and write permissions?**

2. **Is there any restriction to when students may register for and drop classes?**

3. **Do we need to prevent users from manually navigating to pages using the URL bar? As well, since sessions are linked over all sites hosted on gwupyterhub, do we need to confirm on our pages that not only is a session login variable set, but that it is set to a value contained in our database?**

4. **If a grad secretary enters a grade for a student before the faculty instructor, does the faculty instructor still have the opportunity to modify it once?**

5. **Is it possible for a course to be cotaught by professors? Or to have different professors per section of the same class?**

Draft schema:

//for all student accounts (graduate students)
//contains the student (university) id (uid) -- primary key
//the first and second address lines as you'd enter for a billing address
//first and last name, middle initial excluded but entirely possible to add
//their password (generated by us, not by them -- by requirement)
//their email
//the program in which they are in (either master's or PhD)
//the year in which they were admitted into the university
//their degree
Table student{
uid int(8) [PK]
address1 varchar(80)
address2 varchar(80)
fname varchar(40)

```
  lname varchar(40)
  password varchar(25)
  email varchar(40)
  admission varchar(4)
  degree varchar(20)
  program varchar(30)
}

//a duplicate of the student table, but dedicated only to students who have graduated or
no longer attend the university
Table paststudent{
  uid int(8) [PK]
  address1 varchar(80)
  address2 varchar(80)
  fname varchar(40)
  lname varchar(40)
  program varchar(30)
  password varchar(25)
  email varchar(40)
}

//dedicated solely to the “faculty instructor” role
//uid is still university id
//address stored although unlikely to be required, we can keep it for “complexity”
//password generated by us
//email
Table instructor{
  uid int(8) [pk]
  address1 varchar(80)
  address2 varchar(80)
  password varchar(25)
  email varchar(40)
}

//dedicated solely to grad secretaries and system admins
//uid
//address unlikely to be required
//password generated by us
Table supervisors{
  uid int(8) [pk]
  address1 varchar(80)
  address2 varchar(80)
  password varchar(25)
```

```
  permission varchar(20)
  email varchar(40)
}

//table dedicated to classes being taken by students. If a class is not taken by anyone,
//there are no deletion anomalies because classes are stored in the course table below.
//each record is unique by the uid of the student taking the class, and the crn of the class
//uid is the user id of the student taking the class
//crn is an autoincremented value set up in the course table. It must be entirely unique
//between course and department, and will be since it is generated in the course table.
//semester is simply either “Fall”, “Spring”, or “Summer”
//year is any valid year (maybe a range of 100-200 years)
//grade is the grade received by the student
//gradeModified is simply a check to see whether the faculty instructor has added the
grade at all, and if so, the grade cannot be modified further.
Table enrollment{
  uid int(8) [PK]
  crn int [PK]
  semester varchar(8)
  year varchar(4)
  grade varchar(12)
  gradeModified boolean
}

//course is the full course list, where extended attributes of each course are listed in
//separate tables to reduce redundancy
//crn is the autoincremented value, which makes it entirely unique among all classes
//section is simply the section number. Since there may be multiple sections, only the
//unique data per section is listed in the course table, and all other information is stored
//in the coursedata table
Table course{
  crn int [pk, increment]
  day varchar(5)
  time varchar(10)
  instructorid int(8)
  location varchar(25)
  section int
}

//duplicate data from multiple sections of a single course within the course table is
//reduced here. Anything that is identical across multiple sections of the same class is
//instead stored here. When collecting data on a course, if looking to report all classes of
//one cid, just join course and coursedata on the crn, look for department and cid
```

```
//crn is the key linkage between course and coursedata
//cid is just the int value (just course number) for a given course
//dept is the department in which the course belongs to.
//name is the course name
//credits is the credit hours
Table coursedata{
  crn int [pk]
  cid int
  dept varchar(40)
  name varchar(40)
  credits int(2)
  semester varchar(8)
}

//as a note, location is not just the building, but also the room number. Each building in
//combination with a room number is unique
//capacity is just the number of students who may register for the class
Table room{
  location varchar(40) [pk]
  capacity int(4)
}

//prereqs is an isolated list of prerequisites based on specific class crn. This allows
//for the prevention of needing to have large duplicated records of courses where only
//the prerequisite list changes. Since a class may have multiple prerequisites, and a class
//may be a prerequisite for multiple classes, both items must be primary.
Table prereqs{
  crn int [pk]
  prereq int [pk]
}

//identical concept to the prereqs table
Table coreqs{
  crn int [pk]
  coreq int [pk]
}

//same as prereqs
Table lab{
  crn int [pk]
  lab int [pk]
}
```