Final Report - CourseCraft

Surreal Engine - Nikita Belausau, Ryan Coslove, John Hoban, Dorian Hobot, Nick Makar

1. **Implementation Report**

Requirement	Implementation Plan	Progress	Comments
F1	**Database that includes user login info, and a login page.**	***100%***	
F2	**A separate navbar on the homepage for each type of user, dean,prof,student. Personal info is queried from database and displayed**	***100%***	
F3	**Create a course registration method in dean navbar that adds new classes to the database.**	**100%**	
F4	**Create a course registration method in professor navbar that registers that professor as the teacher for a specific class in the database.**	**100%**	
F5	**Create a course management page with creating announcements method, create assignments method, and an input grade method in the professor navbar.**	**100%**	

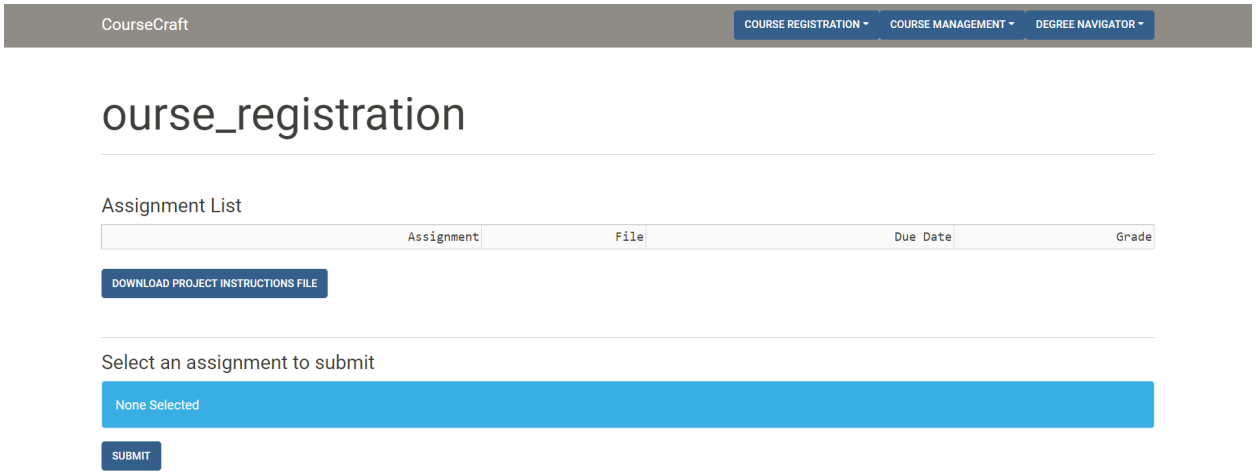
	This will add announcements assignments and grades to the database		
F6	**Create a course registration method in student navbar that allows the student to register for a course. This will add their info to the classes database**	**90%**	**Course registration doesn't account for time conflicts.**
F7	**Create a view schedule method in student navbar that allows students to view their schedule based on their classes, by querying the DB**	**100%**	
F8	**Create a dynamic navbar in the current students homepage that displays all their classes, when you click on a class it will have the ability to view all course contents**	**100%**	
F9	**Create a semester gpa and future gpa calculator method in the student navbar.**	**100%**	
F10	**Create an emailing system when specific conditions are met**	**100%**	
NF1	**Hidden password, password hashing, ensuring that ids are unique**	**85%**	**We did not implement hashed or encrypted passwords.**

NF2	**Host the system on a server that is accessible 24/7.**	**0%**	**We did not host CourseCraft on a persistent server.**
NF3	**Auto-increment/checking for duplicates**	**100%**	
NF4	**Ensure that our system works for as many people as we can manage, which is 5, then just extrapolate that information to see if it'd work for 1000**	**20%**	**It works for multiple users but hasn't been thoroughly tested.**
NF5	**Test on mac, windows, linux**	**66.7%**	**Tested on mac and windows.**

2. **Build and Run Instructions**

- Clone the CourseCraft repository: `git clone https://github.com/ryanclove/CourseCraft.git`
- Follow the instructions under Installing Miniconda and Creating the Environment (Windows) on this Confluence page.
- From the Miniconda terminal, run `conda activate ccenv_windows`
- Once the conda environment is active, run `python web.app` from inside the `CourseCraft/` directory.
- View `http://127.0.0.1:8050` in your browser, which should show CourseCraft.

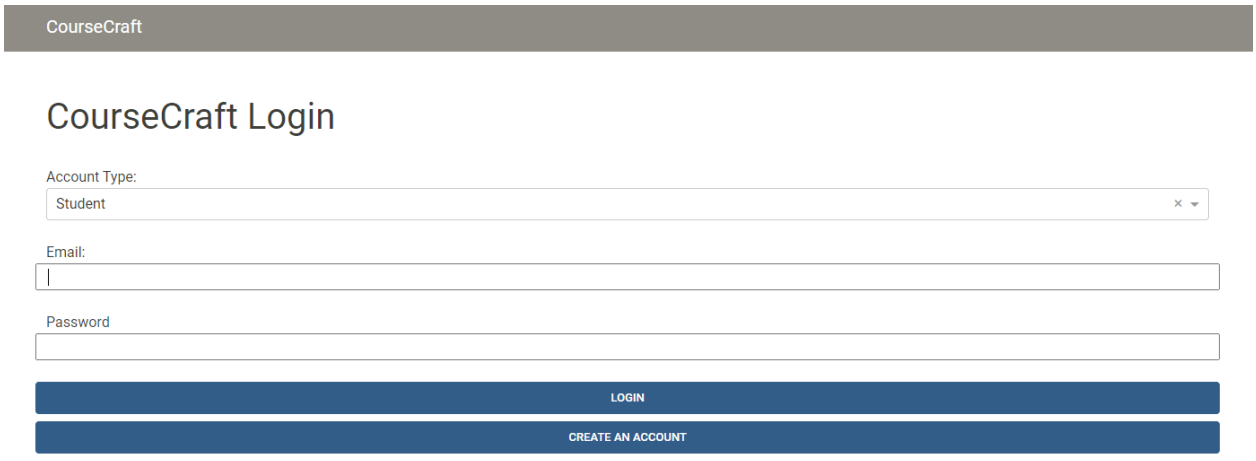
3. Bug Report



- Switching from a course to Course Registration page causes an incorrect render.
- Refreshing on any page returns the user to the login screen.
- A successful login shows a login error briefly before redirecting the user to the proper homepage.
- A user remains enrolled in a class if it is deleted from the database. However, it is not possible to delete a class as a front end user.
- Students course credits are not initialized to 0 on creation.

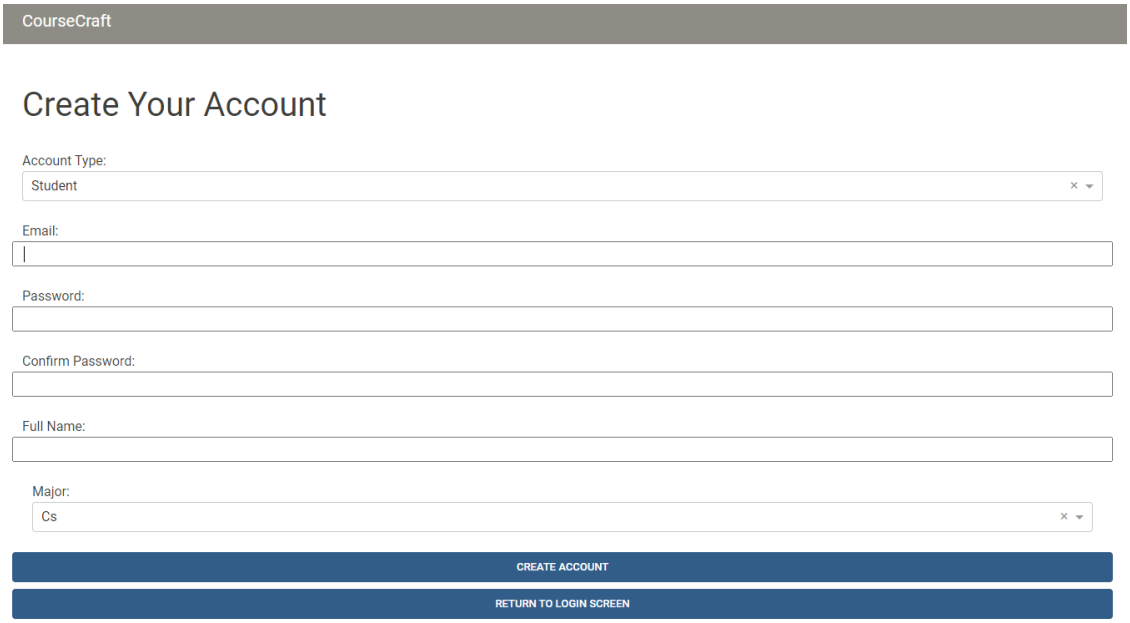
4. Screenshots

Login Page



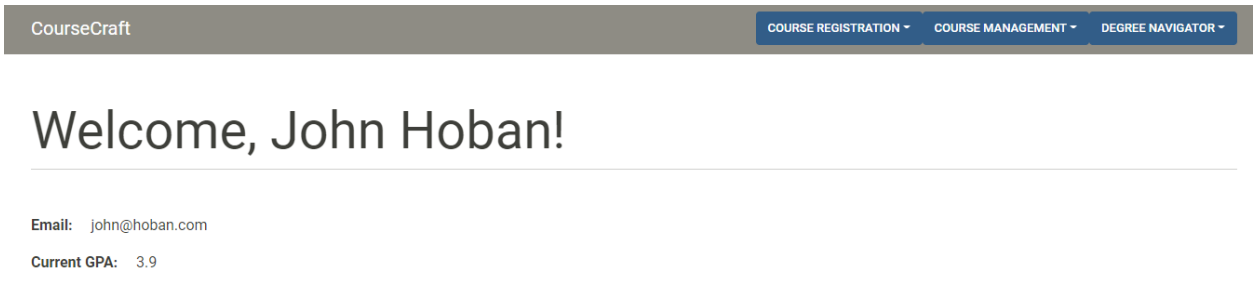
This is the frontpage of CourseCraft and the first thing you should see. Here you should choose a role and enter the email address and password associated with your CourseCraft account, then hit submit to proceed. Conversely if you do not have an account click the create account button to create one.

Create Account Page



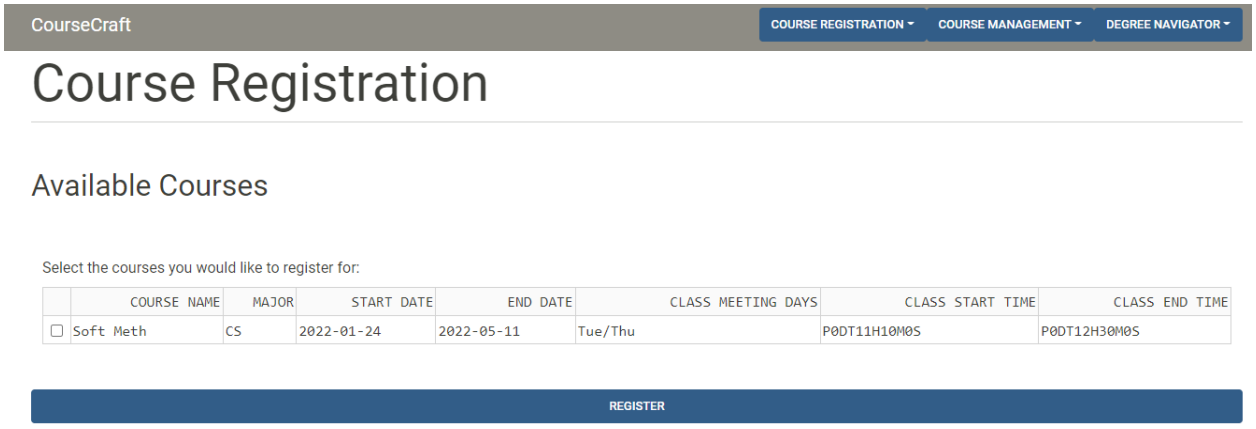
On the create account page you pick a role, enter your email address, a password, name and choose a major, then click create account to create a CourseCraft account so you can access CourseCraft services.

Student Front Page



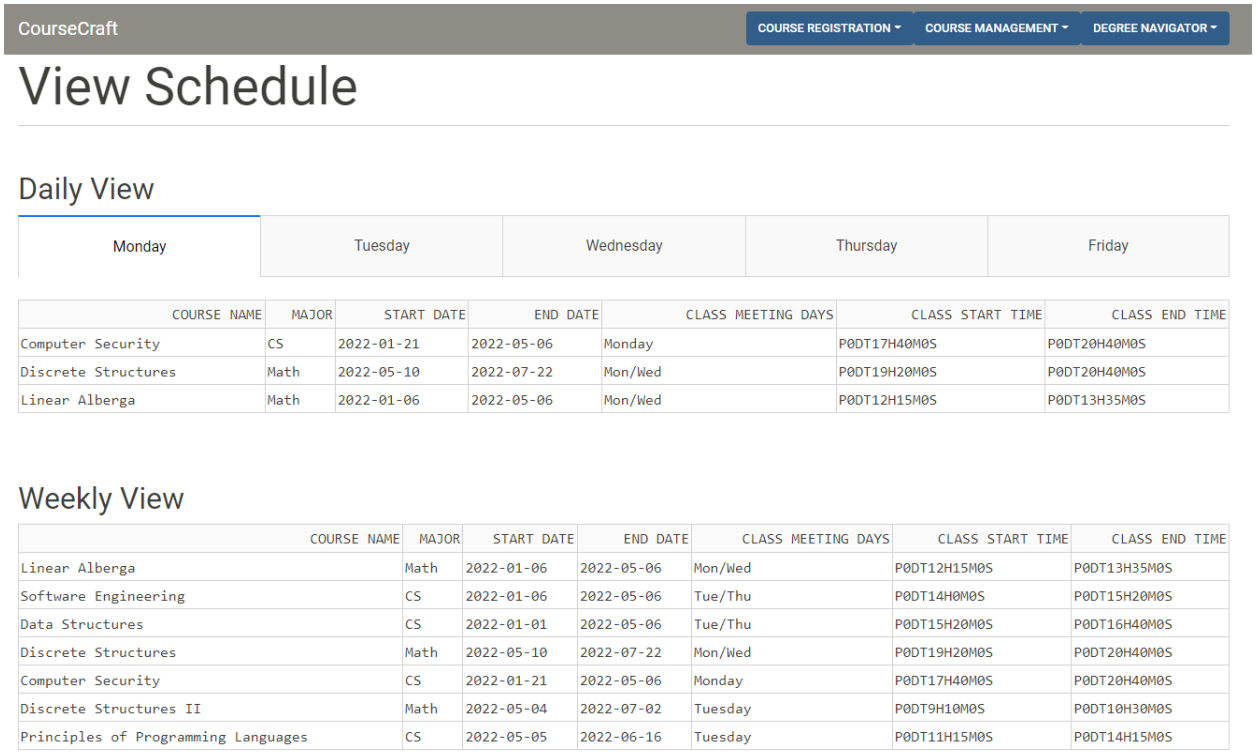
If your account is associated with the student role then you should be redirected here upon login. This page shows some user information, and lists all the student services available in the navigation bar.

Course Registration



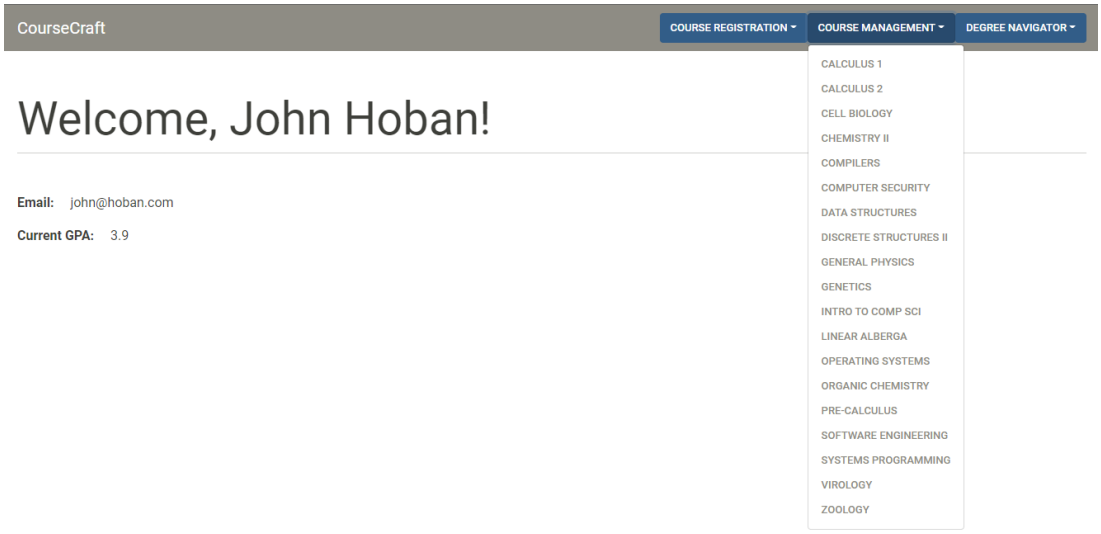
You'll be redirected here if you select register for a class from the navbar drop down menu under course registration. On the course registration page you'll get a list of all available classes, you can select one or multiple checkboxes and hit submit to enroll in those classes.

View Schedule



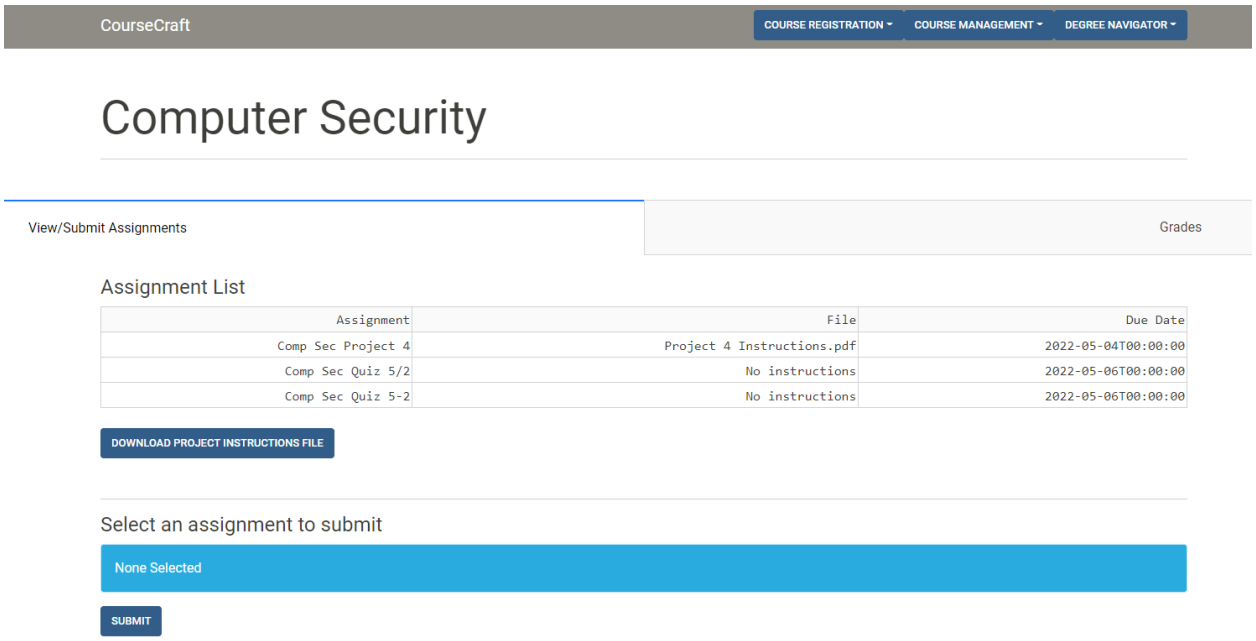
Allows you to view the weekly and daily schedule of classes you have enrolled in . You can click on the week day tabs to see what classes you have on that day.

Course Management



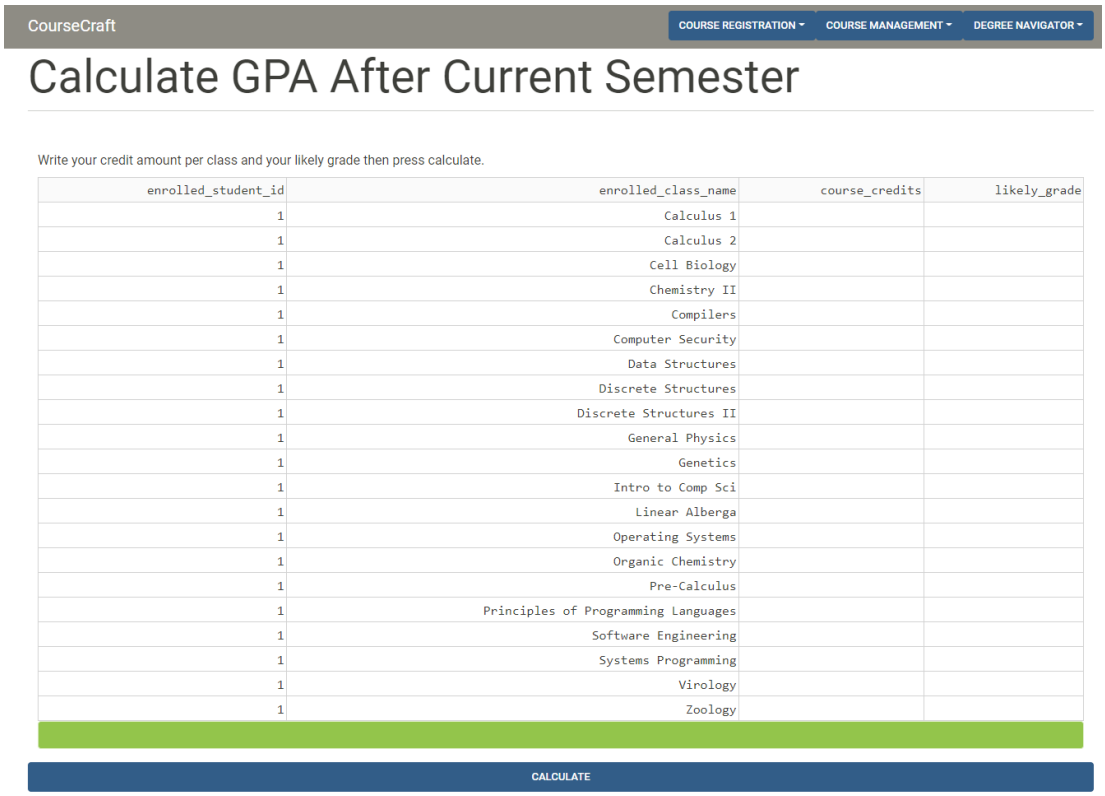
Clicking course management on the navigation bar will list all classes that you are registered for. Clicking on one of the classes in the drop down menu will bring you to a course page for that specific class.

Course Pages



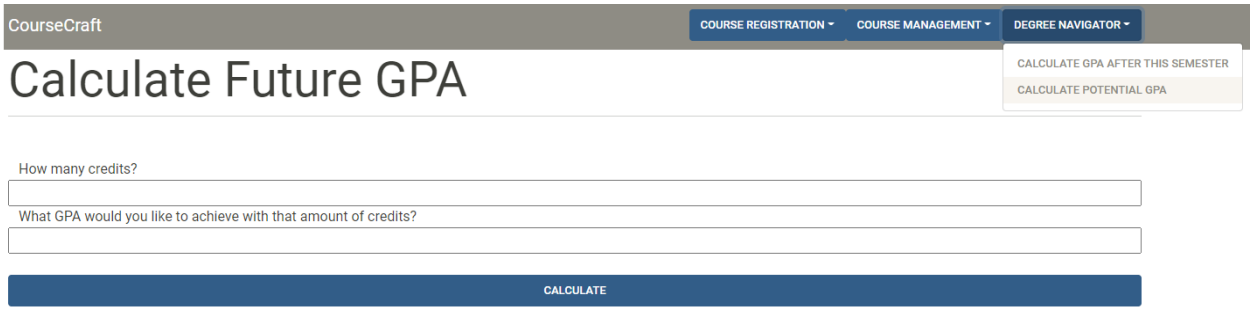
On the course page you can enter 1 of 2 tabs. The first tab is where a student can view any assignments and its due dates for a particular class. You can submit an assignment by clicking on the corresponding row of the assignment in the table and selecting a file from your local drive by clicking submit. You can also download the assignment by clicking on a row and hitting download.

Calculate GPA After This Semester



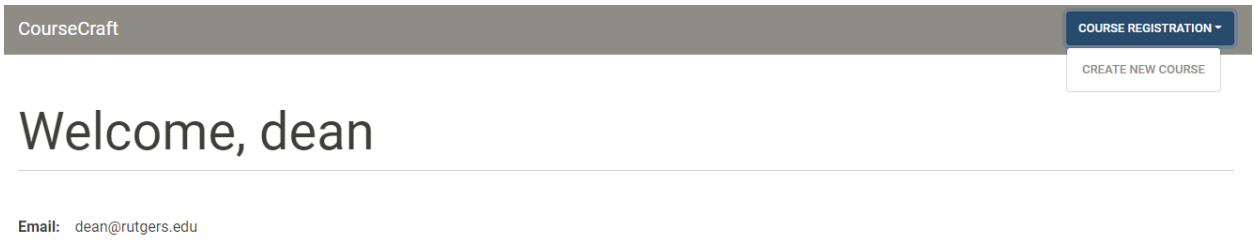
On this page you can check what GPA you will get based on entering hypothetical grades for each class you are currently taking. This takes into account already taken credits and your current GPA.

Calculate Future GPA



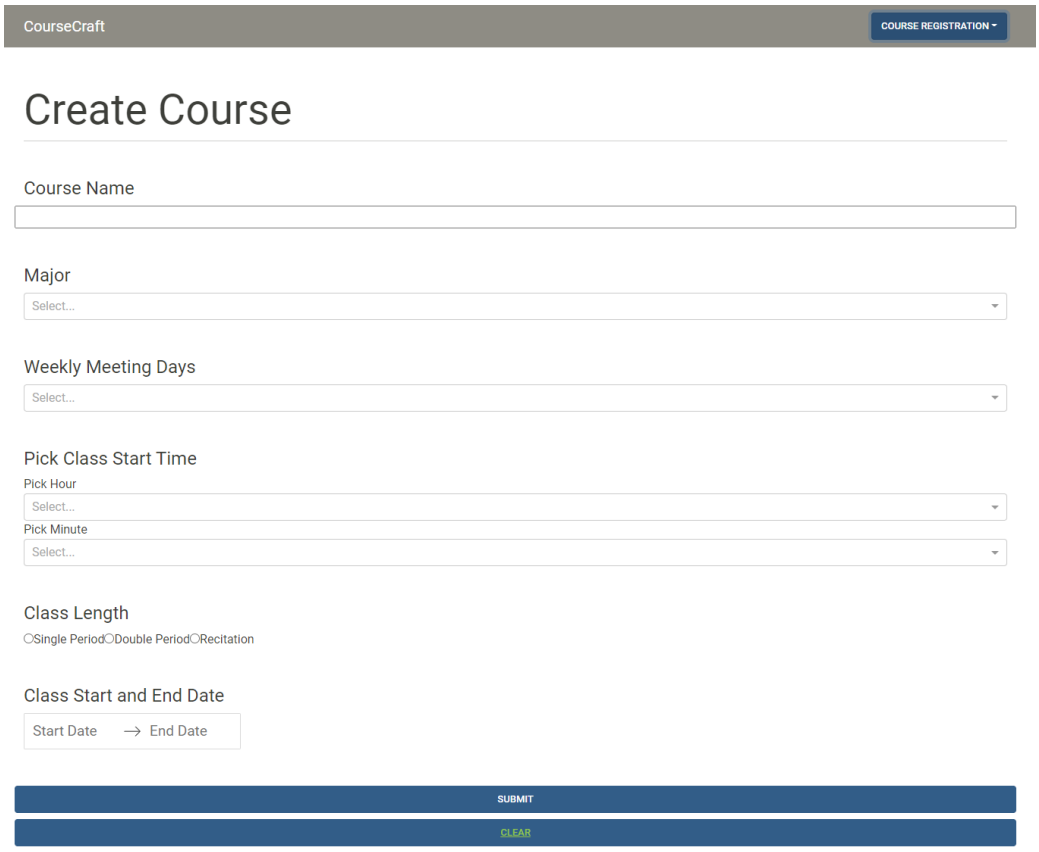
On this page you can check if a desired GPA is possible by taking a hypothetical amount of credits. This takes into account already taken credits and your current GPA and will tell you the average grade needed for the amount of hypothetical credits taken to achieve the desired GPA.

Dean Front Page



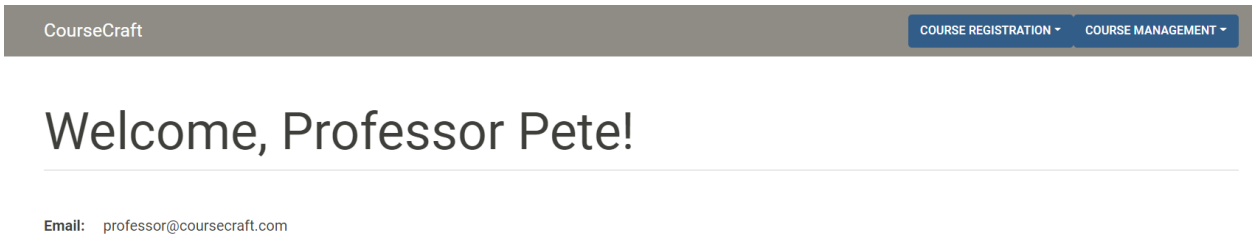
If your account is associated with the dean role then you should be redirected here upon login. This page shows some user information, and lists all the dean services available in the navigation bar.

Create Course



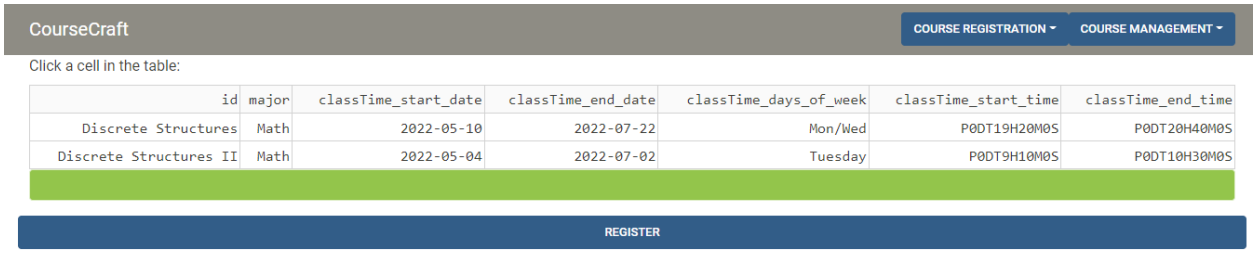
If you selected create course from the dean front page drop down menu then you will be redirected here. This page is used to enter a new class in the database. Enter the class name, major and scheduling information then hit submit to create a new class. You will be prompted if the class was submitted successfully.

Professor Front Page



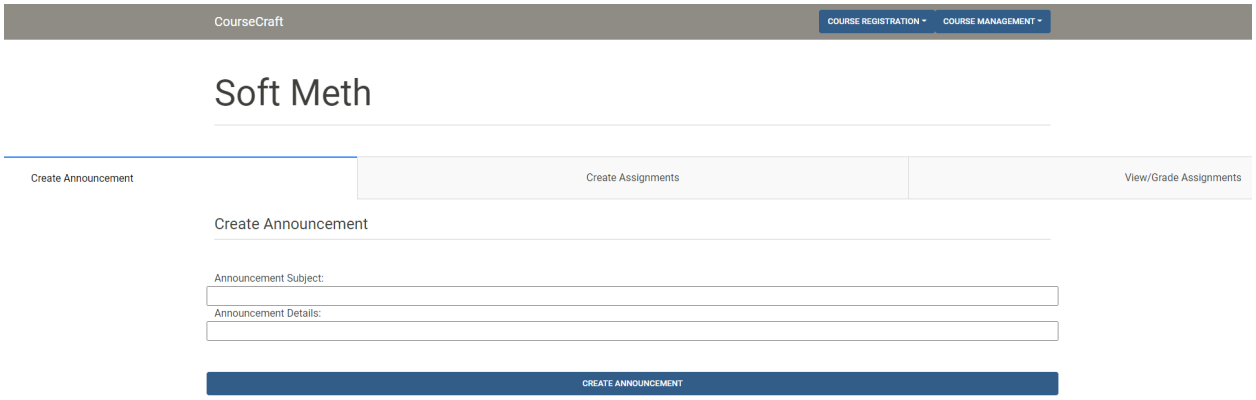
If your account is associated with the professor role then you should be redirected here upon login. This page shows some user information, and lists all the professor services available in the navigation bar.

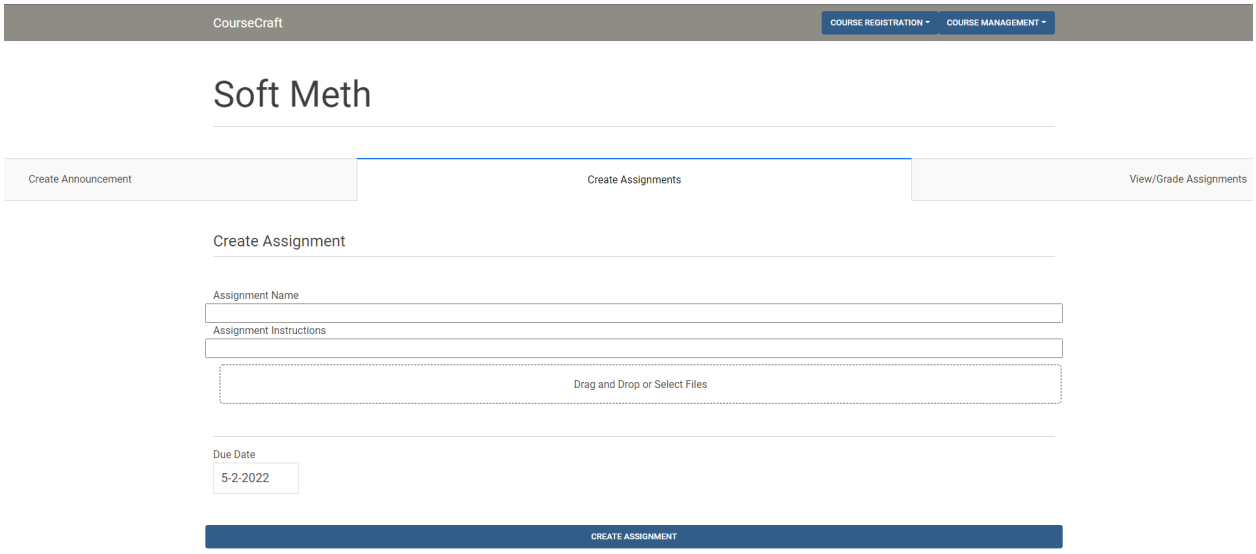
Register to Teach A Class

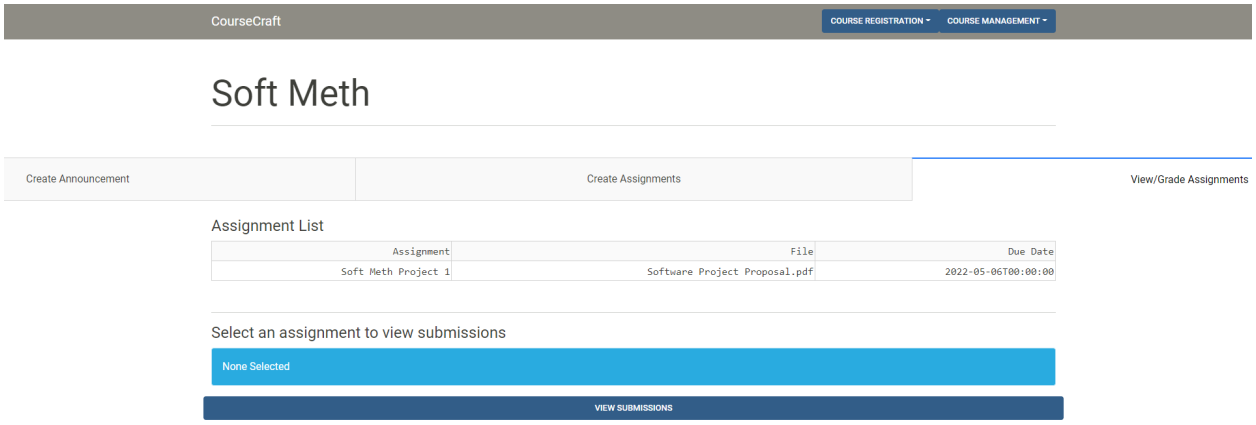


This page lists classes that do not yet have a professor assigned to teach them, from here you can click on the row of the class you'd like to teach and then click register in order to sign up to teach the class.

Course Management







If you click the course management drop down in the navigation bar, you'll get a list of classes that you're registered to teach. You can then click on one of the individual classes in the drop down menu to go to that courses page. The course page contains 3 tabs for each of the professor's class functionality. A professor can create an announcement in the Create Announcement tab. A professor can add an assignment by entering the Create Assignment tab, filling out the create assignment form, and clicking submit. The professor can also view a list of the assignments in the course and choose to see any submissions posted by students by selecting a row from the assignment list and clicking view submissions. The professor can then choose a row with a student's submission and enter a numeric grade for that student's submission.

5. Project Metrics

	Auto-Generated	External Source	Written Manually	Total
Lines of Code	0	8987	3035	12,022

	Source Files	Classes	Modules
Total	20	7	15

	External Modules	Built-in Modules
Total	7	8

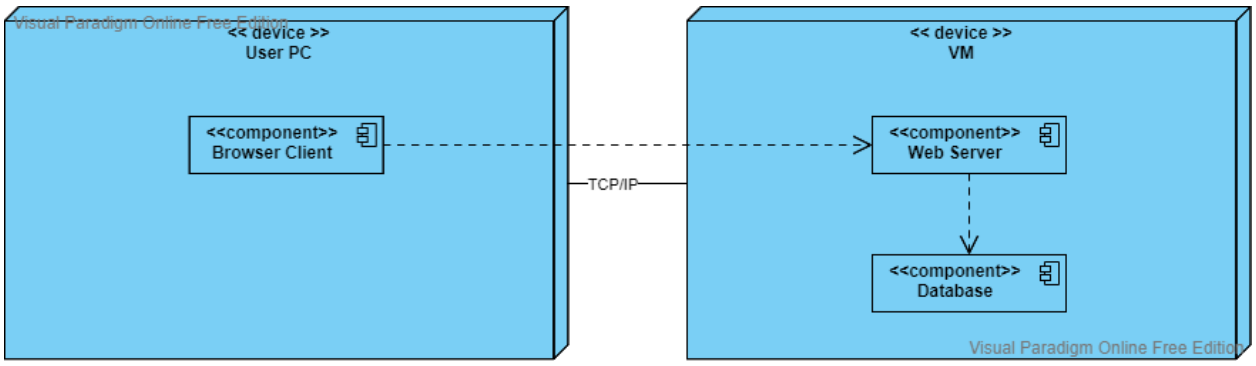
External Modules	Built-in Modules
dash dash_bootstrap_components dash_html_components dash_table pandas pytest	datetime base64 functools typing email smtplib enum

Module	Lines of Code
datetime	2524
base64	603
functools	992
typing	2675
smtplib	1140
enum	1053

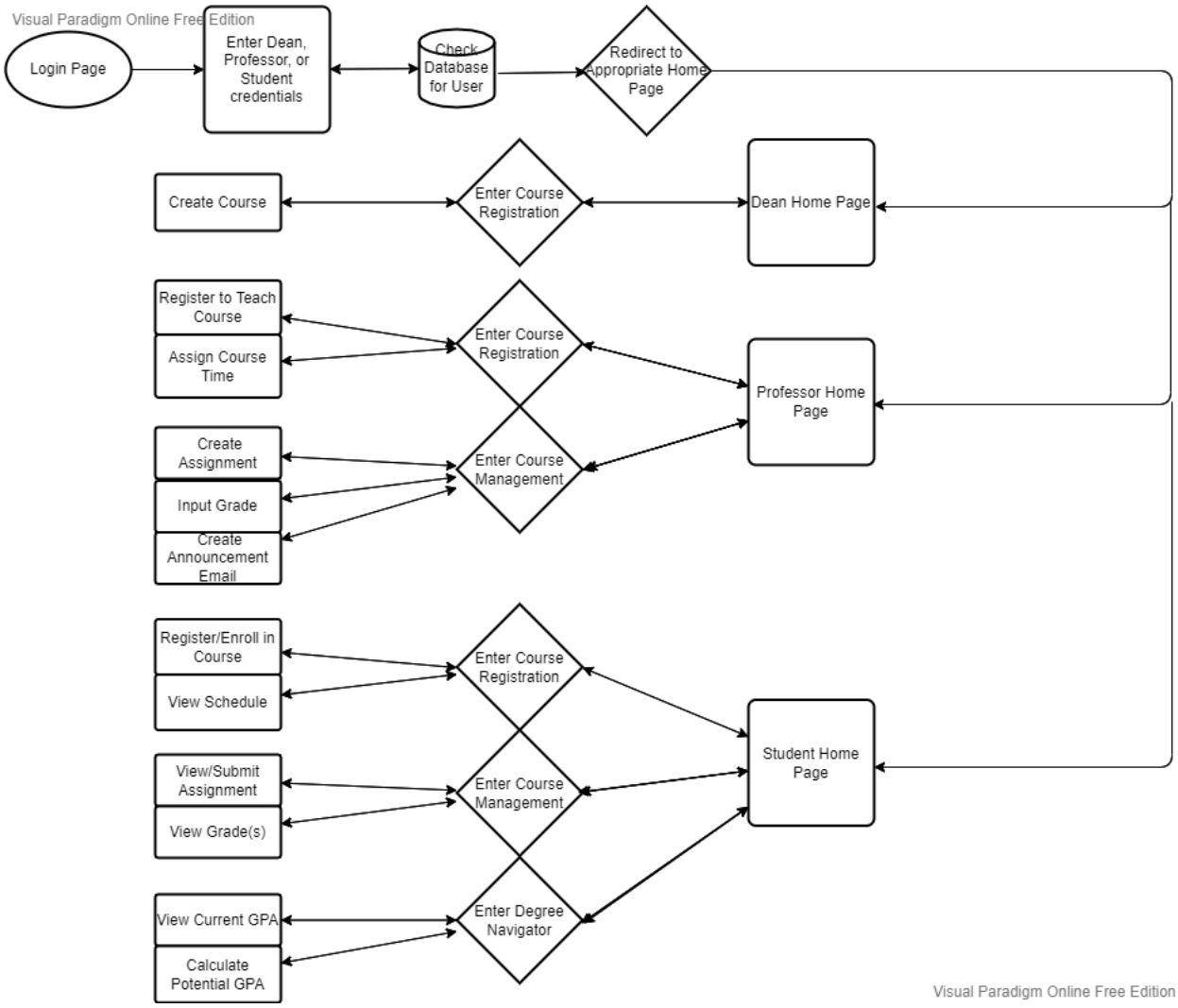
Class or Source File	Containing File (if Class)	# of Functions/Methods	Lines of Code
ccapp.py	-	0	6
create_account.py	-	2	155
dean.py	-	2	172
lecture.py	-	5	222
login.py	-	3	146
main.py	-	2	54
navbar.py	-	2	105
not_found.py	-	0	4
prof_lecture.py	-	11	475
professor.py	-	3	104
student.py	-	10	509
ccemail.py	-	7	197
dbconnector.py	-	27	373
user.py	-	0	52
DBConnector	dbconnector.py	27	373
UserType, Major, User, Student, Professor, Dean	user.py	0	52
test_dean.py	-	3	28
test_ccemail.py	-	15	179
test_professor.py	-	7	60
test_student.py	-	7	61
test_calc_future_gpa.py	-	5	38
test_calc_semester_gpa.py	-	6	95

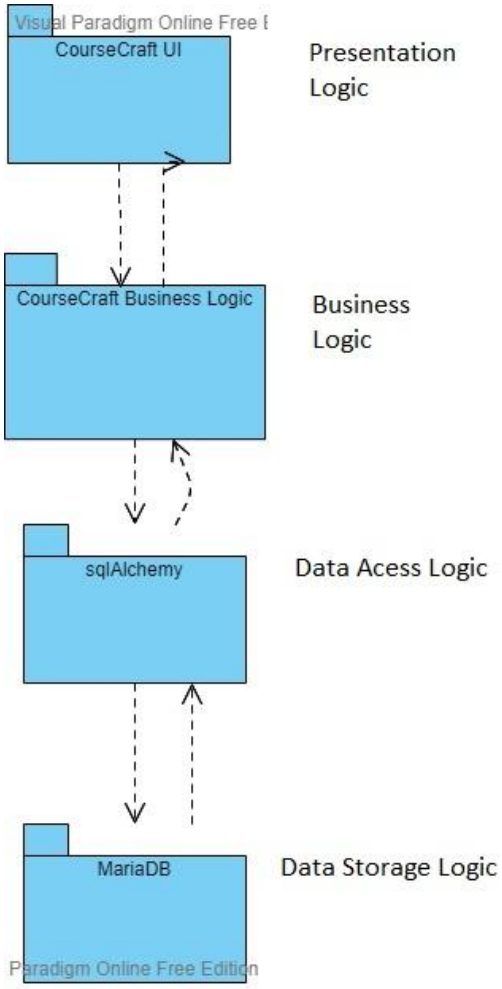
6. Software Structure and System Architecture

Our Software Architecture is Client/Server style:



User interface:





Our System Architecture is also detailed at the following google doc: [System Architecture]
Our System Models are also detailed here in our SRS: [SRS Doc]

7. Summary

Our initial plan was to build a course management system, called “CourseCraft”, to combine all activities related to course scheduling and administration into a single online tool. Webreg and Canvas are two important tools at Rutgers University, and other universities use similar software. We wanted to combine their fragmented functionalities into one program to offer a more centralized resource and better user experience. We wanted to add features to allow students to be more informed and prepared when selecting their classes for an upcoming semester, as well as provide warnings for course selections, assignments, upcoming exams, current academic performance, etc. Course administrators would be able to provide SPNs through the site and organize their pages in a user-friendly and clean format. Our plan was to allow Deans to create courses, Professors to teach and manage those courses, and for Students to enroll in those courses to participate as well as allow students to schedule their classes. We also planned on allowing our system to alert students of new announcements, grades, and assignments created by their professors.

We were able to create most of our intended requirements. As intended: deans can create courses; professors can register to teach courses; professors can create assignments, announcements, and grade assignments; students can create and view their course schedule; students can enter their course pages and view and submit assignments; students can receive emails notifying them of a new announcement for their class, a new grade change for their submitted assignment, and when successfully submitting their assignment; Students can calculate their current GPA as well as their future/potential GPA. What we did not implement was the SPN feature we had discussed in the initial project proposal. However, we did complete a centralized resource that allows deans, professors, and students to interact in a user-friendly platform. There were not significant changes between our initial plan and the final product. We created 13 user-stories and were able to accomplish all 13, plus additional tasks.

Contributions:

Ryan Coslove:

As the Product Owner, my contributions for the project were to host the software project on my github, create the SRS template, dynamically create the course pages for professors, full-stack implementation of professors being able to: create assignments, create announcements, grade assignments. Within grading assignments the professors are able to download the submissions from students. I also worked on the students being able to retrieve/download the instructions from the assignments the professors created. Finally I reformatted the student and professor lecture pages so they would include tabs and be cleaner-looking. I also added necessary queries to our database connector so the virtual machine was updated with accurate data. I personally enjoyed this project a lot. I was fortunate to be with great team members who were helpful and readily able to communicate. A lesson I learned to implement as a future software developer is better time management. I often found myself doing the bulk of my work late in the sprints, often on or slightly before the last day of the sprint. So in the future, allocate my time more efficiently.

John Hoban:

As the Scrum Master, my role alongside developing CourseCraft was to help plan sprints alongside the Product Owner and to help my teammates's progress if they were having difficulty. Most of my effort was in helping them with Dash early in the project, since I had the most experience with this framework of anyone on the team. Outside of managing scrum, I was responsible for implementing the login page, the database connector class, and the email system. It was enjoyable to work on these parts since they all contribute to a different part of our system, so I got to be involved in a wide scope of features for our project. Overall, I think our end result was successful, but we definitely underestimated the difficulty of the project as a whole. It was easy to propose features in the design phase, but implementing them was an entirely different story. Still, we included all of the originally planned features and had decent test coverage for them. Going forward, some lessons I'll keep in mind are how to effectively coordinate work among several team members, how to create requirements that are easily verifiable and not too abstract, and creating an extensible software design from the start of the project so that adding new features is easy.

Mikita Belausau:

As a Software Dev, my contributions for the project were the original database creation, the professors registering to teach courses front-end and back-end, and the GPA calculations for both semester and possible future GPA front-end and back-end as well. I'm pretty happy with how the project turned out, everyone did a great job. I enjoyed looking at everyone's contributions and how they thought to implement their respective parts, it definitely gave me nice influences for my own parts. I think for the future I need to balance my time a bit better in the future, and update my work being done on jira more consistently.

Dorian Hobot:

As a Software Developer, my contributions to the project have been the dean page and all dean services which included the creation of classes, initial implementation of lecture.py which dynamically creates pages for each class for the student roll and allowed viewing and uploading of assignments, and I created the home pages for each role Student, Professor, and Dean. The project was a bit more challenging than expected, but I think we achieved a decent result and everyone in the group contributed well. I think it was a good experience to create a functioning website and I think I'll find use for what I learned in the future.

Nick Makar:

As a Software Developer, my contributions to the project have been creating the initial student functions, testing professor and dean functionality and implementing all code inspection changes. The initial student functions included creating a course registration page and a student schedule page. The implementation for both of these features changed over the course of the project to become more user friendly and to adapt to changes we implemented throughout our sprints. These features allowed students to register for multiple courses and view their schedules on a daily or weekly basis. The testing for professor and dean functionality mainly focused on testing the creation of classes, registering for courses, creating assignments and accessing assignments from the database. The project was more difficult than expected due to learning new tools and time constraints, but I feel that we were able to submit a decent web app. Overall I think that we worked well together and I look forward to using the skills I learned throughout this project in the future.