San Francisco State University
Final Project for SW Engineering Class CSC 648-848
Section 05 Fall 2022

TEAM #6

ReqCheck

Names	Role	Emails
Alex Sanchez	Team Lead	asanchez26@mail.sfsu.edu
Victoria Wilson-Anumudu	GitHub Master	vwilsonanumudu@mail.sfsu.edu
Syed Faiz	Back End Lead	sfaiz@mail.sfsu.edu
Eric Falk	Front End Lead	efalk1@mail.sfsu.edu
Vivek Santoki	Front End	vsantoki@sfsu.edu
Erik Rodriguez	Front End	erodriguez7@mail.sfsu.edu

http://ec2-52-53-211-193.us-west-1.compute.amazonaws.com/

December 15, 2022

Table of Contents

Product Summary

Function	Priority	Actors	Function
4	1	Application	The application should protect the encapsulated user data and disclose information based on roles.
43	1	Course	Each course shall be given a category of major, minor, GE, or elective.
1	1	Degree Planner	To use the degree planner feature, students must fill the appropriate subjects that they want to take for their degree.
37	1	Rec Checker	The Req checker shall be available on the WWW platform.
57	1	System	System will be available during all hours of the day including weekends and holidays.
25	1	System	The system shall display the course requirements for graduation.
26	1	System	The system shall differentiate courses by requirements and electives.
5	1	User	The user (student) must be given information on whether a specific course is transferable from a certain school.
19	1	Course	Course shall have less dependency to any other entity.
18	1	Student	Students shall have a selection of departments they must choose from.
31	2	System	The system shall inform the student any prerequisites required prior to enrolling in a course.
46	2	System	The system shall give numerical status on the number of semesters left before graduating.
3	2	Timeline	The timeline shall depend on how much units a particular student must complete to obtain their degree.
48	2	Course	Each course shall have an enrollment status based whether a grade has been given or not.
42	2	Requirement	Each requirement change shall update the overall course completion right away.
17	2	Admin	Admin shall add the prerequisite on a defined hierarchy of courses.
20	2	Admin	Admin shall have a list of all the courses.
21	2	Admin	Admin shall have a list of courses taken by a student.
6	2	Application	The application should automatically verify a student's academic records via the administrator.
7	2	Application	The application should be able to authenticate user identity by the university.
51	2	Course	Each course shall display the semester’s it is being offered for.
52	2	Semester	Each semester shall display the courses that can be taken prior to enrollment.
58	2	Student	Students shall be authenticated by the university to use the system.

28	2	System	The system shall update all courses based on students input data.
29	2	System	The system shall inform students of course exemptions.
32	2	System	The system shall update any prerequisite requirements.
34	2	System	The system shall enforce any prerequisites prior to enrolling in courses.
35	2	System	The system shall require satisfaction of prerequisites to enroll in courses.
36	2	System	The system shall give alternatives to satisfying prerequisites if applicable.
49	2	System	The system shall not allow students to enroll in courses they do not qualify for.
50	2	System	The system shall require students to complete prerequisites to enroll.
30	2	System	The system shall inform the student the number of semesters left to graduate.
14	2	Admin	Admin shall set new records and shall make a hierarchy of prerequisite courses.
15	2	Admin	Admin shall arrange all the courses department wise.
45	2	Exemption	Each exemption shall be based on course requirements and student grade level.
53	2	Main Page	The main page shall display the user's schedule and classes with professor’s name, date, and class number.
12	2	Professor	A professor shall fill up a seat once a student had dropped the course.
10	2	Professor	Professors shall only approve a defined number of students to their courses.
11	2	Professor	A professor shall fill in dropped students’ seats using a waitlist of students.
22	2	Student	Students shall see the list of professors for a course.
13	2	Student	Each student shall have a chance to request enrollment and each enrollment seat shall be filled up by a first come first serve basis.
33	2	System	The system shall be notified if a course has not been passed.
23	2	System	The system shall display the student’s grade level (current, transfer, new).
24	2	System	The system shall inform students of any changes in department requirements.
41	2	System	The system shall give a grade level status based on overall course completion.
27	2	System	The system shall notify students of all newly required courses.
47	3	Course	Each course shall have numerical options of perquisites.
44	3	Course	Each course shall be appointed based on a student’s grade level (current, transfer, new).

8	3	System	System shall provide all information on passed courses, professors, grades, and sessions.
16	3	Admin	Admin shall change any courses/prerequisites as per the demand of the admin panel.
61	3	People	The amount of people shall be defined by the college administration.
9	3	System	System shall send notifications of recommended courses and waitlisted courses for users to add.
59	0	N/A	
2	0	N/A	
60	0	N/A	
38	0	N/A	
39	0	N/A	
40	0	N/A	
54	0	N/A	
55	0	N/A	
56	0	N/A	

Title: ReqCheck

Since most colleges have their own course curriculums for their programs, students find it quite difficult to make these transfers due to discrepancies between their courses. It typically takes about 2 to 3 weeks to finalize/approve a transfer due to demand and a lack of cohesivity between school programs and course codes. Therefore we built this software Req Check, where we typically solve this problem by strategically creating value to save time and provide knowledge to both students and school administrators on how to handle transfers.

Uniqueness

The Req Check software takes one school's course code and matches it with the other schools' courses. An automated system like Req Check will efficiently map out when a student will graduate as well as ensure they are on the right track as a new transfer student. Req Check will alleviate stress for students and faculty as well as save time and money. At the end of the day, the goal is to help both parties and stakeholders plan and handle the transfers beforehand seamlessly.

URL to your product accessible to instructor, on deployment server:

http://ec2-52-53-211-193.us-west-1.compute.amazonaws.com/

Milestone Documents (M1-M4)

San Francisco State University
SW Engineering CSC 648/848 Fall 2022
Req Check
TEAM #6

Names	**Role**	**Emails**
Vivek Santoki	Team Lead	vsantoki@sfsu.edu
Victoria Wilson-Anumudu	GitHub Master	vwilsonanumudu@mail.sfsu.edu
Alex Sanchez	Front End Lead	asanchez26@mail.sfsu.edu
Syed Faiz	Back End Lead	sfaiz@mail.sfsu.edu
Eric Falk	Front End	efalk1@mail.sfsu.edu
Erik Rodriguez	Front End	erodriguez7@mail.sfsu.edu

Milestone 1

Milestone Version	Date
M1V2	October 19, 2022
M1V1	September 15, 2022

TABLE OF CONTENTS

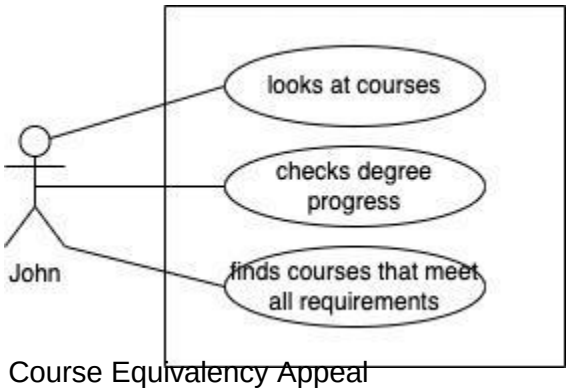
Executive Summary

Students across North America oftentimes find the need to transfer between schools throughout their university career for various reasons. Unfortunately, since most colleges have their own course curriculums for their programs, students find it quite difficult to make these transfers due to discrepancies between their courses. It typically takes about 2 to 3 weeks to finalize/approve a transfer due to demand and a lack of cohesivity between school programs and course codes. This is why we built this software Req Check, where we typically solve this problem by strategically creating value to save time and provide knowledge to both students and school administrators on how to handle transfers.

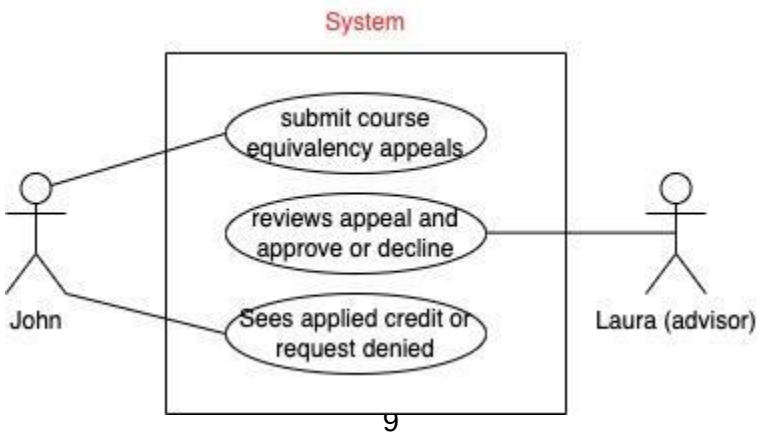
The Req Check software takes one school's course code and matches it with the other schools' courses. An automated system like Req Check will efficiently map out when a student will graduate as well as ensure they are on the right track as a new transfer student. Req Check will alleviate stress for students and faculty as well as save time and money. At the end of the day, the goal is to help both parties and stakeholders plan and handle the transfers beforehand seamlessly.

Main Use Cases

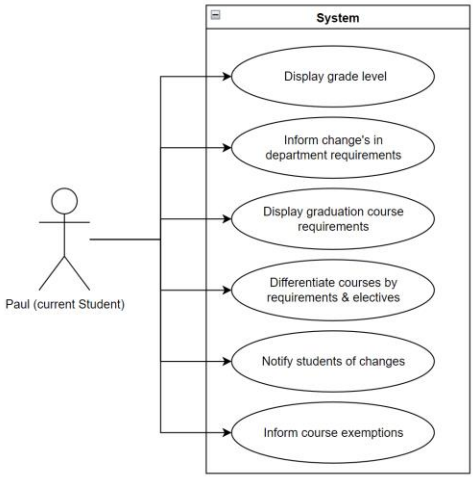
Use Case: Efficient Degree Planning
Actors: John (Student)
Description: John is a computer science student at SFSU and is excited to graduate. He is missing several requirements including Area B2, American Ethnic and Racial Minorities, Social Justice, U.S. History, and U.S. Govt. Wanting to graduate as soon as possible, John wants to find a course that satisfies all of these requirements at once. He uses Req Check planner to get a list of the fewest number of classes that will satisfy all of his requirements. John discovers that HIST 471 will satisfy all of his outstanding requirements and is thrilled that he can finish his degree by taking only 3 more units.



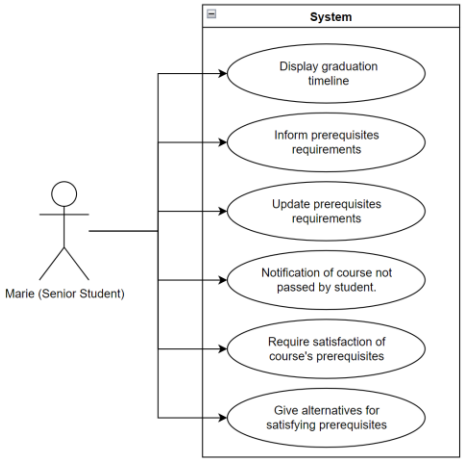
Use Case: Course Equivalency Appeal
Actors: John (Transfer Student), Chris (Transfer Student), Laura (Advisor)
Description: John, a transfer student from UNLV is having a hard time deciding which courses will transfer over to SFSU from his previous university. He logs into Req Check and sees that many of his courses do not have existing equivalencies at SFSU. He has taken calculus I and calculus II at his old school and believes they are equivalent to MATH 226 and MATH 227. Using Req Check, John starts an equivalency appeal and submits the syllabi from his previous math courses for review. Laura reviews the syllabi, and seeing the obvious similarity, grants the equivalency. When another student, Chris, transfers from UNLV the following year, he is happy to see that SFSU has automatically accepted his calculus courses.



Use Case: Students Exempted from Department Course Requirements
Actors: Current Student (Paul), Courses
Description: Paul is a declared current CS student at SFSU and is halfway through his academics. Paul received an email from the CS Department informing him of a change in requirements for graduation. The course csc520 will longer be required for graduation, and will be an elective. Furthermore, an additional course, csc317 is now required for newly declared CS majors and transfer CS students. Paul decides to drop the course csc520 and enroll in csc317. However, Paul is exempted from taking both courses since the department course requirements will be required for new or transfer students only.

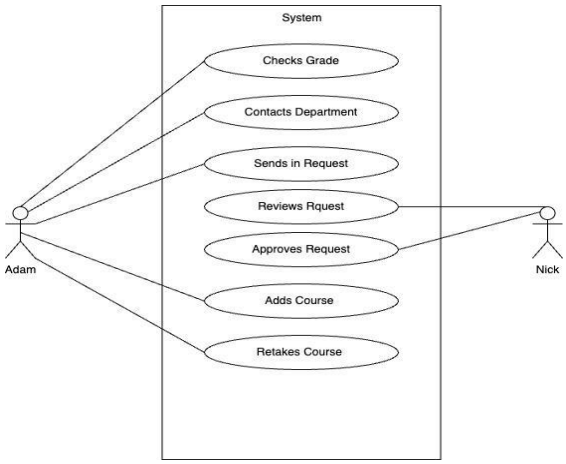


Use Case: Students satisfying prerequisites for electives
Actors: Senior Student (Marie), Prerequisite
Description: Marie has one last semester left to graduate from SFSU as a computer science major. She is currently taking a required class in the spring semester in order to satisfy the prerequisite for an elective in her final semester of fall; however, she did not pass the course. The elective has a prerequisite that will be required and enforced, so she will not be able to take the elective in the final semester. However, the required class will be taught in the summer and if Marie is able to enroll and pass, she will be able to satisfy the prerequisite.



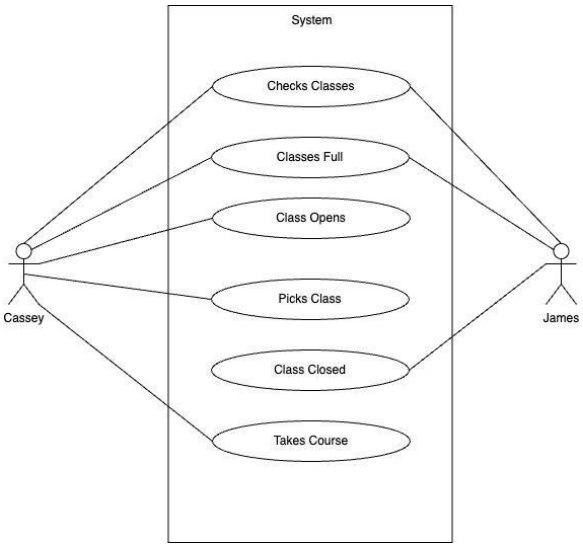
Use Case: Grade Check
Actor: Adam(Current Student) and Nick(Professor)
Description: Adam had recently taken MATH324 and received a C- but he needs at least a C to continue on to his next course, which would be CSC 415. Unfortunately, the current system will not let him retake the course since the school system deemed him a passing student and will not allow him to retake the course despite him needing a C or better to take CSC 415. Adam has to go through the department to help fix the issue, and until it is resolved, he will not be able to continue with most of his important core major classes unless he retakes MATH324 for a better grade. Adam will also need multiple signatures and permission from his previous MATH324 teacher, professor Nick, the dean, and the department in order to retake the course.

Use Case Diagram:

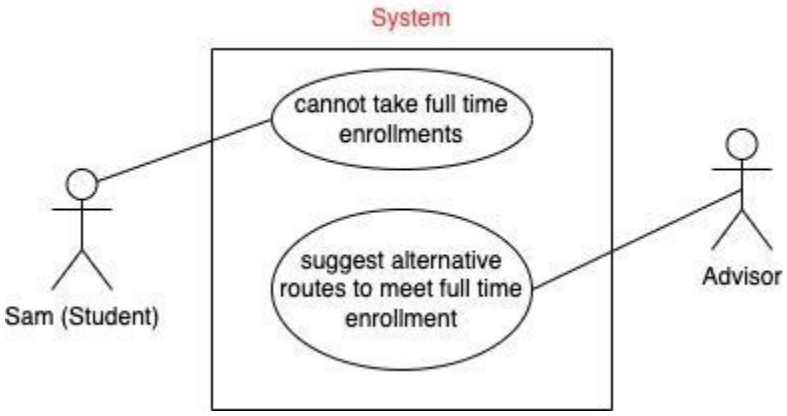


Use Case: Course Notification
Actors: Casey(Current Student), James(Current Student)
Description: Casey is looking to add courses for his fall semester at SFSU, but unfortunately, all the classes he wants to take are full. James also wants to take the same courses as Casey, but the same classes are full. Casey gets lucky because he checks later that day to find two spots open and adds the course, but James is too late to add the course. If only there were a system in place that let the student know if a seat was available in the course instead of having to check manually at random hours of the day.

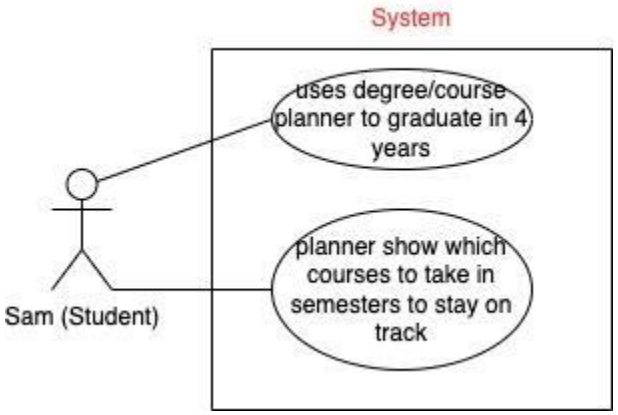
Use Case Diagram:



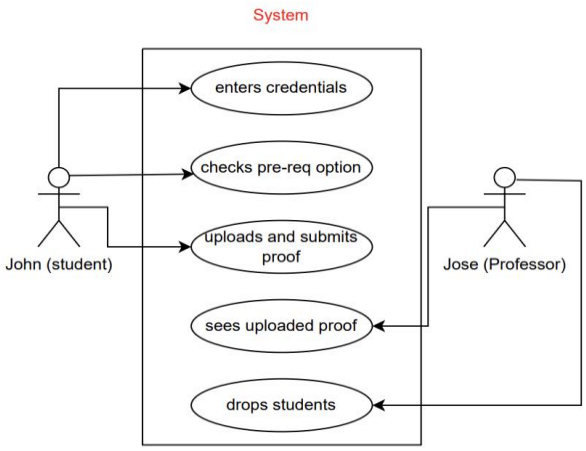
Use Case: Semester courses are offered
Actors: Sam (Transfer Student), Advisor
Description: Sam is a new transfer student and is trying to plan her fall semester. She wants to enroll as a full time student this year which is a minimum of 12 units. She has completed all of her general education requirements and is having a hard time making a full time schedule due to the fact that certain courses are being offered only in the spring. She cannot take certain electives to make her a full time student because of certain courses that did not transfer over from her previous university.



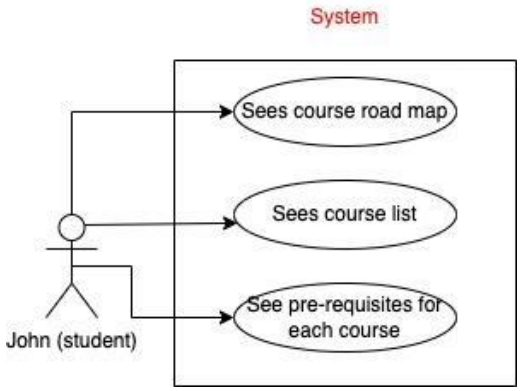
Use Case: Fastest route to graduation
Actors: Sam (Transfer Student), Advisor
Description: Sam is trying to figure out which courses to take that will keep her on track to graduate in four years. She is having difficulty figuring out which courses to take each semester because certain courses are only offered in specific semesters. The current degree roadmap that she is using is not clear about which courses are offered in a certain semester.



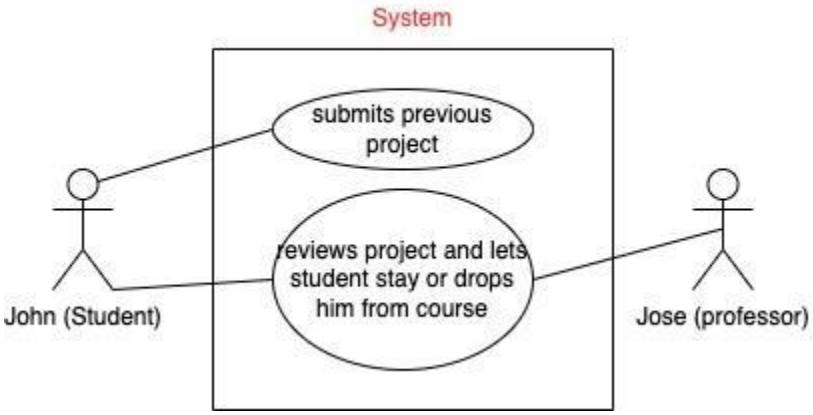
Use Case: Check Students Prerequisite.
Actors: Jose (Professor), John (Student).
Description: Jose is a professor of computer science at SFSU and teaches many courses each semester. All students at SFSU taking his courses must satisfy prerequisite requirements to stay in the course or get dropped. Jose's job is to check each student's pre-requisite status individually which is time consuming and inefficient. To save Jose time and redundancy, all students will self-report pre-req status for the course. This will save Jose time and hassle of manually checking for pre-req for 100s of his students.
Use Case Diagram:



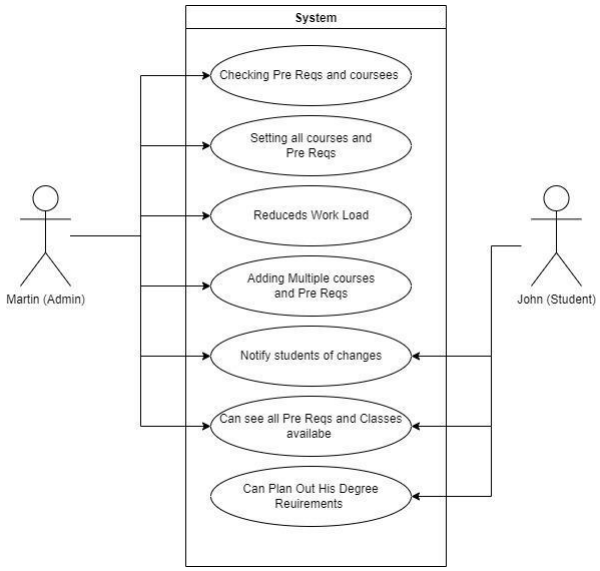
Use Case: Prerequisite Road Map Chart
Actors: John (Student).
Description: John is a student of computer science at SF state and wants to know what classes he needs to take to stay on track and graduate in a timely fashion. However, the SF website has information about the computer science degree program and the computer science department only has information about the prerequisite chart. There is no single page that provides complete information and clarity on which courses to start with and follow along. This new roadmap will provide John information about what course he needs to take and can take to stay on track, so he does not have to be confused and bounce back and forth between two webpages.
Use Case Diagram:



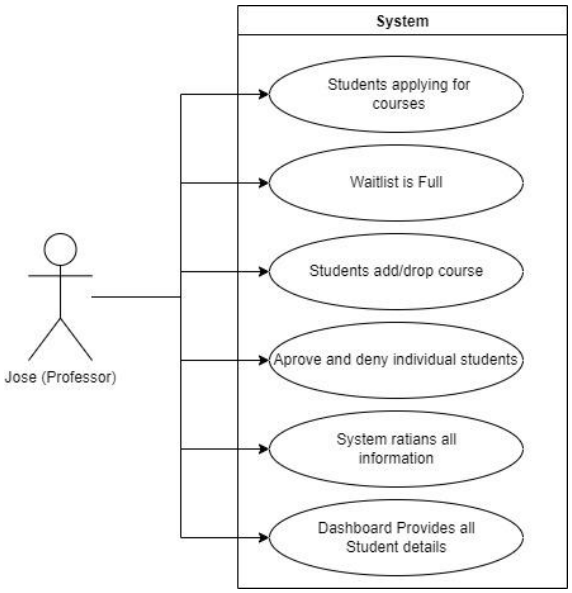
Use Case: Accepting or Rejecting to the course.
Actors: Jose (Professor), John (Student).
Description: Jose is the professor and John is a student. Jose wants to accept the student but wants to know a little bit more about his work for a particular project.John will give the idea about the course work while he requests the professor to include it in his course.The way Jose will have more idea about the student background and related work done by John and it gets easy for Jose to accept or reject the John in particular course.



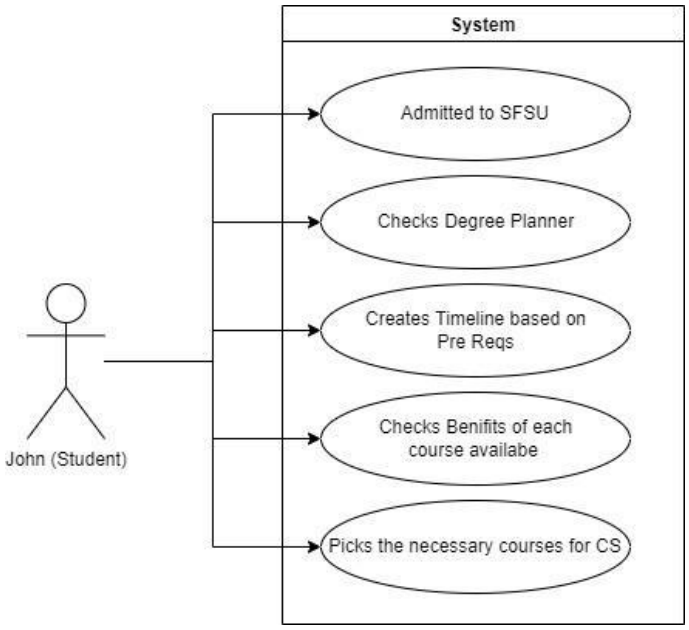
Use Case: To set the Prerequisite courses.
Actors: Martin (College Course Admin), John (Student).
Description: As an administrator, Martin handles all the courses and prerequisite courses of the program. Using req check, Martin easily sets all the courses and their prerequisite courses reducing Martin's administrative work.He can add multiple prerequisite courses and can also add non prerequisite courses. This will help John to see all the prerequisite courses and plan the degree accordingly.



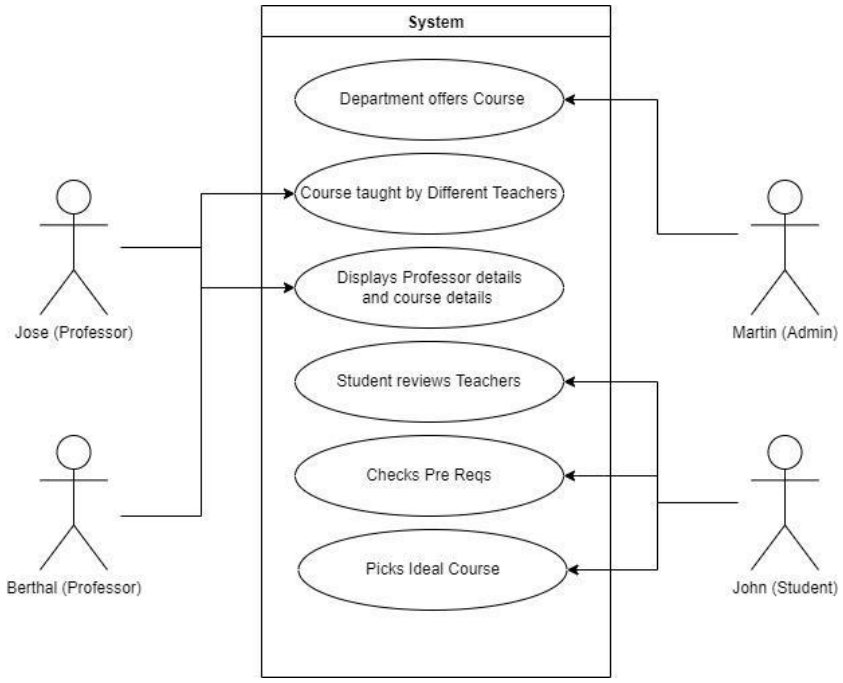
Use Case: Details about the Waiting list and Dropped class.
Actors: Jose (Professor), John (Student).
Description: Many students apply for the same course and it gets difficult to handle all the numbers for Jose for the waiting list, the number of students dropped the class,How many student he approved and their details, to short all the things req checker will help him to provide all the details on his dashboard as a professor.



Use Case: Set Timeline for the Degree.
Actors: John (Student).
Description: John as student he got the admission in SFSU in CS. John wants to know about how much time will take to complete his degree.If John have prerequisite course and he have to take it before his major courses so it will give idea to John about his degree panning.By this way John will have idea about the timeline according that John can take the courses and also can check other courses and their benefits to take the course or not depending on the prerequisite courses.



Use Case: Consistency to add the Prerequisite Courses.
Actors: Martin (College Course Admin), Jose(Professor) Bethel(Professor), John(Student).
Description: Jose is professor and Martin is course admin.In order to keep the maintainability about the prerequisites course for each department, Each course has to be set exactly the same prerequisite course across the Jose and Bethel who is teaching the same course to the students.Across all the courses and departments will have the same manner to check the prerequisite and John will have the better idea about the whose professor course he will choose.



List of Main Data Items and Entities

Student:
First Name
Last Name
Student ID number
Student email
Student cell

Professor:
First Name
Last Name
Employee number
Professor email
Professor cell

Course:
Course Number
Course Title
Units/Credit
Semester
Location

Records:
Transcripts
Student ID

Road Map:
Course Number
Course Title
Semester

Enrollment:
Credits
Course
Student ID
Space

Grade:
Letter
Student ID
Course Number
Course Title
Semester

Transfer Credits:
Institution Name
Course Number
Course Title
Units/Credits
Student ID

Equivalency Appeal:
Appeal ID
Outcome

Notification:
Email
Course
Section

Attendance
Present
Absent
Total Count

Waitlisted Student
Student first name
Student last name
Student ID #
Email

Dropped Students
Student first name
Student Last name
Student ID #
Email

Deadlines
Last day to add
Last day to drop

Request
Request ID
Request type
Request description
Request timestamp
Optional link

Student Courses
Course Number
Course Title
Course Description
Course Credit

Syllabus
Course
Description

Prerequisites List
List of Courses

Initial list of functional requirements

1. In order to use the degree planner feature students have to fill the appropriate subjects that they want to take in the degree.
2. Degree planner shall suggest to them the timeline of the degree where students can have an idea about how much time they need to complete all the courses and prerequisite courses in order to complete the degree.
3. Suggestion of the timing shall depend on how much credit a particular student has to complete their degree.
4. The application should protect the encapsulated user data and disclose information based on roles.
5. The user (if student) must be given information on whether or not a specific course is transferable to a certain school.
6. The application should automatically verify a student's academic records hence the administrator.
7. The application should be able to authenticate user identity by the university.
8. System shall provide all information on passed courses, including professor, grade, and session.
9. System shall send notifications of recommended courses and waitlisted courses for users to add.
10. Professor only can approve a defined number of students to their courses.
11. Once any student dropped the seats shall be filled up by the professor on the waiting list queue of the students.
12. Once any student drops, the professor shall have the idea about filling up that seat.
13. Each student shall have a chance to approve a request but seats shall be filled up by the first come first serve.
14. Admin can set all the new records and can make a hierarchy of prerequisite courses.
15. Admin can also arrange all the courses department wise.
16. Admin can change any time courses and prerequisites as per the demand of the admin panel.
17. Admin can also add the prerequisite on a defined hierarchy of courses.
18. Students shall have category wise subjects that they have to choose.
19. Course shall have less dependency to any other entity.
20. Admin can have all the list of all courses and how much students have taken the courses.
21. Students can see the timing of classes and if they are classing it shall not be chosen automatically by the system.
22. Students can see the list of all the professors for the same course to get better knowledge about the professor course structure.
23. The system shall display the student’s grade level (current, transfer, new).
24. The system shall inform students of any change in department requirements.
25. The system shall display the course requirements for graduation.
26. The system shall differentiate courses by requirements and electives.
27. The system shall notify students of all newly required courses.
28. The system shall update all courses based on students input data.
29. The system shall inform students of course exemptions.
30. The system shall inform the student the number of semesters left to graduate.

31. The system shall inform the student any prerequisites required prior to enrolling in a course.
32. The system shall update any prerequisite requirements.
33. The system shall be notified if a course has not been passed.
34. The system shall enforce any prerequisites prior to enrolling in courses.
35. The system shall require satisfaction of prerequisites in order to enroll in courses.
36. The system shall give alternatives to satisfying prerequisites if applicable.
37. The Req checker will be available on the WWW platform and a person who is authenticated and respected can use the software.
38. The Website can serve the number of people at a time and the number will be defined by the college administration.
39. As we used an optimized library to build the website so it will give the performance even if the user has low internet.
40. Web sites can have the changes in terms of functionality and design wise but will make sure that all the changes will be found in our user manual if there are any updates in the website.
41. System shall give a grade level status based on overall course completion.
42. Each requirement change shall update the overall course completion right away.
43. Each course shall be given a category of major, minor, GE, or elective.
44. Each course shall be appointed based on a student's grade level (current, transfer, new).
45. Each exemption shall be based on course requirements and student grade level.
46. System shall give numerical status on the number of semesters left before graduating.
47. Each course shall have numerical options of prequisites.
48. Each course shall have an enrollment status based whether a grade has been given or not.
49. System shall not allow students to enroll in courses they do not qualify for.
50. System shall require students to complete prerequisites in order to enroll.
51. Each course shall display semester's it is being offered for up to 4 years.
52. Each semester shall display the courses that can be taken prior to enrollment.
53. Main page will display the user's schedule and classes with professors name, date and class number.
54. All classes that can be selected will be green while classes that can not be taken will be red. If the course being taken is a prereq the following course will show yellow.
55. System will be able to support a large number of users at one time.
56. System will run on Windows, Mac and Lynx.
57. System will be available during all hours of the day including weekends and holidays.
58. Students have to be authenticated by the university to use this tool as a university internal tool with the role of student.
59. The system shall schedule an advising for students when seeking an equivalency appeal.

List of non-functional requirements

1. All types of user will have the credentials that are provided by their university in order to access the req checker.
2. All the professors have to be authenticated by the university to use this tool as a university internal tool with the role of professor.
3. All the admin have to be authorized by the university admin office in order to access this tool.
4. All students(users) after the accepted admission are eligible to access the tool and can view their course curriculum.
5. Every student shall get the information about the subject and their prerequisite courses and also the student shall see if there are any second level prerequisite courses.
6. Every student can choose the subject and can send the request to the professor to let them in their courses.
7. Every student can send the 2 types of request to the professor which shall let them in the course or they can request to eliminate prerequisite courses to eliminate them as per the student experience.
8. Students can share additional information about the work he has done with approval requests to the processor.
9. Students can also request to eliminate prerequisite courses if they have enough work done related to the course.
10. Students have to give mandatory notes with the reason why they want to take the particular subject.
11. Students shall have ideas about all pending requests that they have sent to the professor.
12. Students shall see the result of acceptance and rejection from the professor and can distinguish between the professor if a student has applied the same course from the two professors.
13. Professor shall have the idea about all the requests through the total request table.
14. Professors can directly accept or reject student requests with both types of request.
15. Professor shall see the additional details along with student requests.
16. Professor can see all the numbers like waiting students, enrolled students and more.
17. Professor has all the rights to accept or reject the application of the student according to the details provided by the student
18. Students can use the degree planner feature that shall give them ideas about the time expected to complete the degree.
19. After matching all the credits and all the subjects planner shall show the roadmap to take the courses and can save those to use letter on.
20. Students can see the list of their approved rejected courses by the professor.
21. Users will only choose the website if they want to make the request for the approved courses and can see the degree planner.
22. Every User will have to agree on the term and services with following conditions that will be shown in the format.
23. To report any issue in the website, the user will fill the contact us page with their little summary about their concern.
24. Outside users can have to face legal action against the respective university.
25. In order to use the website student users must have their past academic info in their portal otherwise It can fail to give you the expected data.
26. Students shall be able to see a roadmap of all existing and future courses that must be taken.

27. Student shall pick their courses directly instead of manually inputting a course number
28. Students shall be able to switch between Fall, Winter and Summer courses.
29. Students shall be able to see the list of the subject that is approved by the professor.
30. Student shall be authenticated after he gets admission in the university.
31. Students shall have ideas about all the subjects and grading methods.
32. Students can see all the basic knowledge about the course while going with the particular course.
33. Professor can see the list of the students who took the course and also track the record of getting dropped students.
34. At a time one subject can be a prerequisite for one or more courses.
35. Students can ask more details about the ideal course and course structure to the professor before beginning with classes.
36. In the list students can have more idea about the courses once they decide to choose in different frames in the tool.
37. Professor shall also provide the details about the TA and other general info about their work which attracts students to take the course.

Competitive analysis

Competitor 1: https://www.transferology.com/index.htm
Competitor 2: https://assist.org/
Competitor 3: https://webapps.sfsu.edu/public/classservices/classsearch
Competitor 4: https://www.collegetransfer.net/
Competitor 5: https://www.commonapp.org/

Identity

	Competitor 1	Competitor 2	Competitor 3	Competitor 4	Competitor 5	Our Product
Info	https://www.transferology.com/index.htm	https://assist.org/	https://webapps.sfsu.edu/public/classservices/classsearch	https://www.collegetransfer.net/	hhttps://www.commonapp.org/	N/A
Target Audience	Students who want to compare and transfer courses to another school.	Students transferring from a public California community college to a public California university	The target audience is SFSU Student Transfers and Professors	Students planning to transfer to another college	All type of student	SFSU Students and Faculty
Future Partnership	Yes, it looks like they are primarily dealing with schools on the east coast.	This platform deals specifically with schools in California	There is no future partnership for this company		There is no future partnership with this company	N/A
Perspective from User Point	It looks very simple, straight to the point of adding courses, and getting results.	The website is simple and straight to the point and easy to navigate	The website is very basic and looks to be from the early 2010s with lots of links and tabs	The website appears very clean and modern, and it feels easy to navigate	The website is simple and straight to the point and easy to navigate	N/A

Features

	Competitor 1	Competitor 2	Competitor 3	Competitor 4	Competitor 5	Our Product
Strengths	Has an established network of schools and students through volunteer and services.	Clearly list which course are transferable as well as which courses do not count as upper division credit	You can find specific courses using the customizable search, which shows all courses available for the specific section.	Has an existing database of colleges, known transferable courses, and transfer agreements. Recommends schools based on transferability.	Clarity with course transfer with the ease use cases and nice flow to suggest path	It should have a network of schools, handle most courses and prerequisite cases
Weakness	Manually input transfer course to determine eligibility. Courses restricted to those schools participating.	Does not include a list of transferable general education courses	Takes too long to search for courses, and you have to have other tabs open in order to figure out and pick courses	Does not seem to include every transferable course. Institutions seem to be responsible for manually maintaining transferable courses.	Do not need to evaluate all the included courses and all institution name have the the maintainable transfer courses	Courses restricted to those schools participating
Pricing	No mention of pricing for students, only for schools.	Free to the public.	The website is available for students who are enrolled in SFSU	Free for students, annual subscription for institutions (price not available)	Free for every student	Should be free for students only and have access to the website at any time
Social Media	Facebook and twitter.	There is not any social media present	There isn't a lot of social media presence for this app unless you look for it on the school website. Overall it's difficult to find and isn't advertised on other social media Apps.	Twitter, Facebook, LinkedIn	Linkedin and Facebook	Social media will market towards college and transfer students

Onboarding Experience	You can create a free account, and input courses in order to find out transferability to other schools. Then you have a choice of what schools and be shown all their information.	You do not need to create an account to see which course are transferable	You have to be logged into your SFSU Gateway account in order to access the search engine and website.	Aside from tiles on the home page for commonly used features, there is a Get Started button to guide users through the process of adding transcripts.	It has very smooth on boarding functionality flow with minimum requirement	Use a free account linked to school to input courses easier. You should be able to find all your resources in one place instead of having to find them in multiple places and tabs

Features Table

	Competitor 1	Competitor 2	Competitor 3	Competitor 4	Competitor 5	Our Product
Video Tutorial	++	-	-	-	+	-
Course Search	+	+	-	-	++	-
FAQ List	++	++	-	-	-	-
Inputting Datat	+	-	+	-	+	+

User Interface (UI)	+	+	+	+	-	+
User Experience (UX)	+	+	+	+		++

Feature Exist: + Superior: ++ Does Not Exist: -

Future Improvements

	Advice 1	Advice 2	Advice 3	Advice 4	Advice 5	Advice 6
To be Developed	Visual and statistical roadmap of all courses needed to graduate. Inputting courses that are not in the system and finding transferability.					
To be Improved	Probably adding private schools later on. Also, require individual inputs and transferability.					
Usability (UI/UX)	User interface; students will need a simple way of dealing with course interaction. User experience; must be able to feel a sense of					

	fulfillment in future graduation.					
Social Media	Not really sure besides the usual ones.					
New Market Niche	Probably include non-technical degrees and non-degrees.					

High-level system architecture and technologies used

- The operating system used to develop the software has no restriction but the version restriction will be there for the development environment.
- Team will be using Java Script as the basic development language to develop the software.
- For the front end Team will be using react js as a java script library to build the html pages and in order to make the bundle of the project team will use Webpack.
- To use some ready-made components we will use the material UI to use some html components.
- We will use Node js as a runtime environment to develop the backend server.
- To develop the backend API team will use the express JS framework which will serve the request and will use the middleware for authentication.
- To save the application data, the team will use the mysql database and we will be using the workbench in order to maintain the table and data of the table.
- In order to develop the software, the Team will use the Visual Studio Code to maintain and develop code, To maintain the code of all the project team will be using GitHub.
- In order to host all the front end back end we will use the EC2 server with the AWS services and to handle the request we will use the NGINX.
- Our primary cloud platform server will be AWS, but we will also use Google Cloud Platform. This will be noted in the milestone.
- Our software will be supported by all browsers.

Checklist

1. Team found a time slot to meet outside of the class: DONE

2. Github master chosen: DONE

3. Team decided and agreed together on using the listed SW tools and deployment server: DONE

4. Team ready and able to use the chosen back and front end frameworks and those who need to learn are working on learning and practicing: DONE

5. Team lead ensured that all team members read the final M1 and agree/ understand it before submission: DONE

6. Github organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.): DONE

List of Team Contributions

Part A: Team Lead's evaluation of individual team member's contributions.

Team Lead: Vivek Santoki.

Team Member Name	Role (in project)	Contribution and Rating
Syed Faiz	Backend Lead	8
Alex Sanchez	Frontend Lead	6
Victoria Wilson-Anumudu	Github Master	5
Eric Falk	Front end	5
Erik Rodriguez	Front end	5

San Francisco State University
SW Engineering CSC 648/848 Fall 2022
Req Check
TEAM #6

Names	**Role**	**Emails**
Alex Sanchez	Team Lead	asanchez26@mail.sfsu.edu
Victoria Wilson-Anumudu	GitHub Master	vwilsonanumudu@mail.sfsu.edu
Syed Faiz	Back End Lead	sfaiz@mail.sfsu.edu
Vivek Santoki	Front End Lead	vsantoki@sfsu.edu
Eric Falk	Front End	efalk1@mail.sfsu.edu
Erik Rodriguez	Front End	erodriguez7@mail.sfsu.edu

Milestone 2

Milestone Version	Date
M2V2	November 9, 2022
M2V1	October 20, 2022

TABLE OF CONTENTS

Contents

Data Definitions V2

Student
First Name
Last Name
Student ID number
Student email
Student cell
Course
Section
Level
Status

Professor
First Name
Last Name
Employee number
Professor email

Course
Section
Type
Course:
Course Number
Course Title
Units/Credit
Semester
Location
Section
Prerequisite List

Records
Transcripts
Student ID

Road Map
Course Number
Course Title
Semester

Enrollment

Credits
Course
Student ID
Space

Grade
Letter
Student ID
Course Number
Course Title
Semester

Transfer Credits
Institution Name
Course Number
Course Title
Units/Credits
Student ID

Equivalency Appeal
Appeal ID
Outcome

Notification
Email
Course
Section

Attendance
Present
Absent
Total Count

Waitlisted Student
Student first name
Student last name
Student ID #
Email

Dropped Students

Student first name
Student Last name
Student ID
Email

Deadlines
Last day to add
Last day to drop
Course Hierarchy:
Course Number
Course Title
Semester

Request
Request ID
Request type
Request description
Request timestamp
Outcome

Student Courses
Course Number
Course Title
Course Description
Course Credit

Syllabus
Course
Description

Prerequisites List
List of Courses

Prioritized Functional Requirement V2

Over here expand on the functionalist requirements. In addition, these requirements will be implemented on priority 1, 2 and 3.

Function	Priority	Actors	Function
17	1	Admin	Admin shall add the prerequisite on a defined hierarchy of courses.
20	1	Admin	Admin shall have a list of all the courses.
21	1	Admin	Admin shall have a list of courses taken by a student.
4	1	Application	The application should protect the encapsulated user data and disclose information based on roles.
6	1	Application	The application should automatically verify a student's academic records via the administrator.
7	1	Application	The application should be able to authenticate user identity by the university.
43	1	Course	Each course shall be given a category of major, minor, GE, or elective.
44	1	Course	Each course shall be appointed based on a student’s grade level (current, transfer, new).
47	1	Course	Each course shall have numerical options of perquisites.
48	1	Course	Each course shall have an enrollment status based whether a grade has been given or not.
51	1	Course	Each course shall display the semester’s it is being offered for.
1	1	Degree Planner	To use the degree planner feature, students must fill the appropriate subjects that they want to take for their degree.
59	1	Person	An authenticated person shall use the software.
37	1	Rec Checker	The Req checker shall be available on the WWW platform.
42	1	Requirement	Each requirement change shall update the overall course completion right away.
52	1	Semester	Each semester shall display the courses that can be taken prior to enrollment.
58	1	Student	Students shall be authenticated by the university to use the system.
57	1	System	System will be available during all hours of the day including weekends and holidays.
8	1	System	System shall provide all information on passed courses, professors, grades, and sessions.
25	1	System	The system shall display the course requirements for graduation.
26	1	System	The system shall differentiate courses by requirements and electives.

28	1	System	The system shall update all courses based on students input data.
29	1	System	The system shall inform students of course exemptions.
31	1	System	The system shall inform the student any prerequisites required prior to enrolling in a course.
32	1	System	The system shall update any prerequisite requirements.
34	1	System	The system shall enforce any prerequisites prior to enrolling in courses.
35	1	System	The system shall require satisfaction of prerequisites to enroll in courses.
36	1	System	The system shall give alternatives to satisfying prerequisites if applicable.
49	1	System	The system shall not allow students to enroll in courses they do not qualify for.
50	1	System	The system shall require students to complete prerequisites to enroll.
30	1	System	The system shall inform the student the number of semesters left to graduate.
16	2	Admin	Admin shall change any courses/prerequisites as per the demand of the admin panel.
14	2	Admin	Admin shall set new records and shall make a hierarchy of prerequisite courses.
15	1	Admin	Admin shall arrange all the courses department wise.
19	2	Course	Course shall have less dependency to any other entity.
2	2	Degree Planner	The degree planner shall give a timeline on how much time students need to complete all the courses and prerequisites for a degree.
45	2	Exemption	Each exemption shall be based on course requirements and student grade level.
53	2	Main Page	The main page shall display the user's schedule and classes with professor’s name, date, and class number.
61	2	People	The amount of people shall be defined by the college administration.
12	2	Professor	A professor shall fill up a seat once a student had dropped the course.
10	2	Professor	Professors shall only approve a defined number of students to their courses.
11	2	Professor	A professor shall fill in dropped students’ seats using a waitlist of students.
22	2	Student	Students shall see the list of professors for a course.
13	2	Student	Each student shall have a chance to request enrollment and each enrollment seat shall be filled up by a first come first serve basis.
33	2	System	The system shall be notified if a course has not been passed.
23	2	System	The system shall display the student’s grade level (current, transfer, new).
24	2	System	The system shall inform students of any changes in department requirements.
41	2	System	The system shall give a grade level status based on overall course completion.
27	2	System	The system shall notify students of all newly required courses.

46	2	System	The system shall give numerical status on the number of semesters left before graduating.
3	2	Timeline	The timeline shall depend on how many units a particular student must complete to obtain their degree.
60	2	Website	The website shall serve a limited number of people at a time.
5	3	User	The user (student) must be given information on whether a specific course is transferable from a certain school.
18	3	Student	Students shall have a selection of departments they must choose from.
9	3	System	System shall send notifications of recommended courses and waitlisted courses for users to add.
38	0	N/A	
39	0	N/A	
40	0	N/A	
54	0	N/A	
55	0	N/A	
56	0	N/A	

Initial list of functional requirements

1. To use the degree planner feature, students must fill the appropriate subjects that they want to take for their degree.
    1.1. There will be a table where students can select each course based on semester, division, area, department, etc.
2. The degree planner shall give a timeline on how much time students need to complete all the courses and prerequisites for a degree.
    2.1. There will be a roadmap table for each semester (Fall, Winter, Spring, Summer) and each academic year; this table will show the courses needed to graduate.
3. The timeline shall depend on how many units a particular student must complete to obtain their degree.
    3.1. Based on grade level or courses inputted, the roadmap table will update accordingly.
4. The application should protect the encapsulated user data and disclose information based on roles.
    4.1. This is more technical and will be included.
5. The user (student) must be given information on whether a specific course is transferable from a certain school.
    5.1. When submitting an equivalency appeal, the student will be notified of the appeal decision.
6. The application should automatically verify a student's academic records via the administrator.
    6.1. This should happen, but since we don’t have access to the SFSU system, it’ll be by default.
7. The application should be able to authenticate user identity by the university.
    7.1. This should happen, but since we don’t have access to the SFSU system, it’ll be by default.
8. System shall provide all information on passed courses, professors, grades, and sessions.
    8.1. Should be displayed in the user’s profile.
9. System shall send notifications of recommended courses and waitlisted courses for users to add.
    9.1. This is 2 parts; first, recommended courses are by default. And waitlisted courses notifications are optional when adding courses.
10. Professors shall only approve a defined number of students to their courses.
    10.1. Amount should be created by the professor in their dashboard.
11. A professor shall fill in dropped students’ seats using a waitlist of students.
    11.1. Should take place in the Professor dashboard for each unique course.
12. A professor shall fill up a seat once a student has dropped the course.
    12.1. Should take place in the Professor dashboard for each unique course.
13. Each student shall have a chance to request enrollment and each enrollment seat shall be filled up by a first come first serve basis.

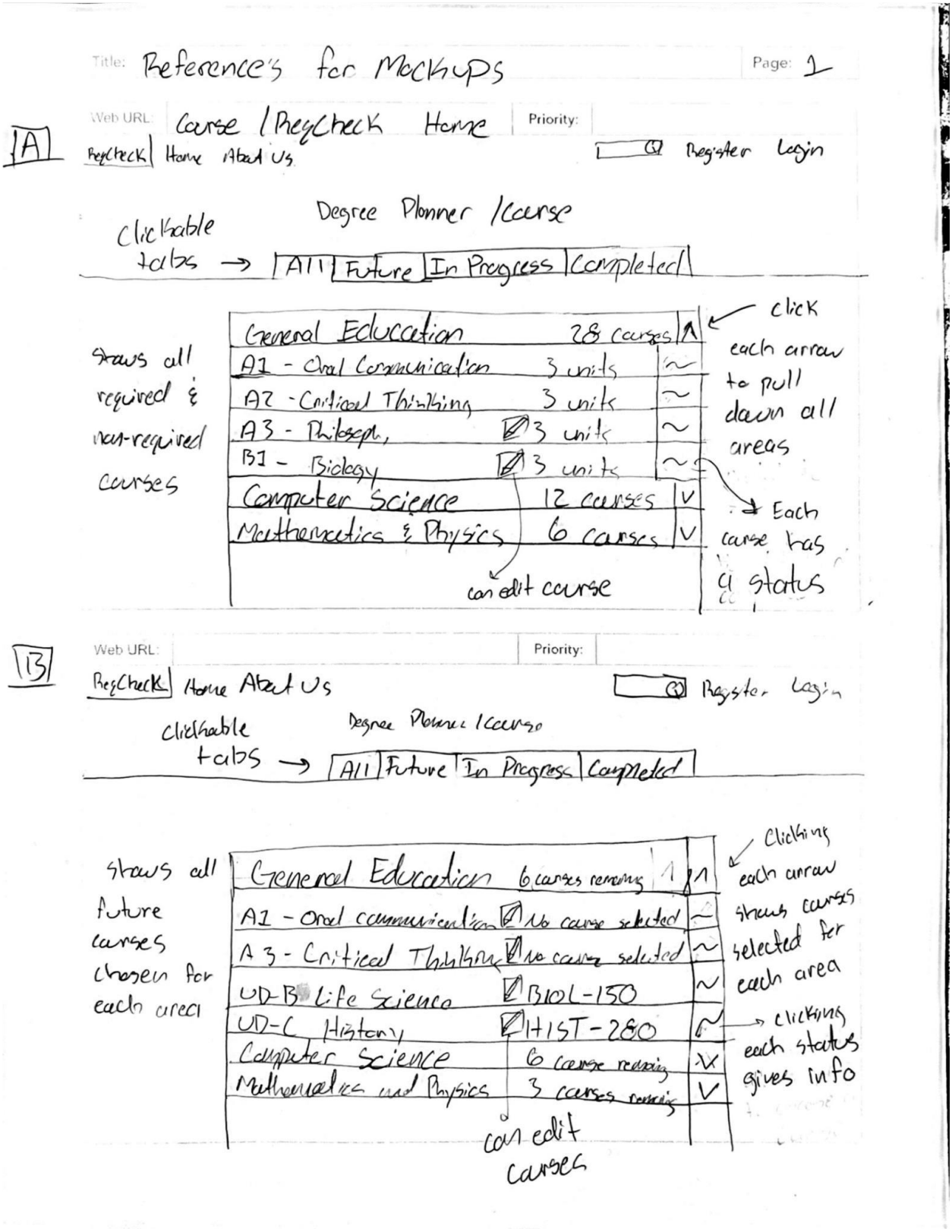
13.1. This is 2 parts; first, it shouldn’t request enrollment if there’s no waitlist. Second, filling a seat is by default a first come first serve. Otherwise, use a waitlist at professors' discretion.

14. Admin shall set new records and shall make a hierarchy of prerequisite courses.
    14.1. Admin will only receive prerequisites approved by all professors teaching the course.
15. Admin shall arrange all the courses department wise.
    15.1. Arrange as in arranging the tables?
16. Admin shall change any courses/prerequisites as per the demand of the admin panel.
    16.1. Only professors could do this, not the admin.
17. Admin shall add the prerequisite on a defined hierarchy of courses.
    17.1. Only professors could do this, not the admin.
18. Students shall have a selection of departments they must choose from.
    18.1. Should be shown in the course page.
19. Course shall have less dependency to any other entity.
    19.1. Not clear.
20. Admin shall have a list of all the courses.
    20.1. By default, yes.
21. Admin shall have a list of courses taken by a student.
    21.1. By default, yes.
22. Students shall see the list of professors for a course.
    22.1. Shown in the course page for students.
23. The system shall display the student’s grade level (current, transfer, new).
    23.1. Shown in the student's profile.
24. The system shall inform students of any changes in department requirements.
    24.1. Notification should be shown in the student's profile.
25. The system shall display the course requirements for graduation.
    25.1. Should be shown in both the Roadmap page and course page.
26. The system shall differentiate courses by requirements and electives.
    26.1. This should be shown in the course page.
27. The system shall notify students of all newly required courses.
    27.1. Should be shown in the student's profile as well as in the course page.
28. The system shall update all courses based on students input data.
    28.1. Inputs should update each student’s overall course table for graduation.
29. The system shall inform students of course exemptions.
    29.1. Based on equivalency courses approved, this should update the overall course table.
30. The system shall inform the student the number of semesters left to graduate.
    30.1. This should be default in the student’s profile and roadmap.
31. The system shall inform the student any prerequisites required prior to enrolling in a course.
    31.1. This should be default when searching for courses.
32. The system shall update any prerequisite requirements.
    32.1. This should be default

33. The system shall be notified if a course has not been passed.
    33.1. This should be shown in the student's profile
34. The system shall enforce any prerequisites prior to enrolling in courses.
    34.1. This should be the default when searching for courses to enroll in.
35. The system shall require satisfaction of prerequisites to enroll in courses.
    35.1. This should be the default when searching for courses to enroll in.
36. The system shall give alternatives to satisfying prerequisites if applicable.
    36.1. This should be the default when searching for courses to enroll in.
37. The Req checker shall be available on the WWW platform.
    37.1. This should be the default.
38. ~~The Website can serve the number of people at a time and the number will be defined by the college administration~~. Moved to non-functional
39. ~~As we used an optimized library to build the website so it will give the performance even if the user has low internet.~~ Moved to non-functional
40. ~~Web sites can have the changes in terms of functionality and design wise but will make sure that all the changes will be found in our user manual if there are any updates in the website~~. Moved to non-functional.
41. The system shall give a grade level status based on overall course completion.
    41.1. Should be default and shown in the student profile and roadmap.
42. Each requirement change shall update the overall course completion right away.
    42.1. Should be the default.
43. Each course shall be given a category of major, minor, GE, or elective.
    43.1. Should be the default and be in the course page.
44. Each course shall be appointed based on a student's grade level (current, transfer, new).
    44.1. Should take in consideration all course inputs, equivalency appeals, etc.
45. Each exemption shall be based on course requirements and student grade level.
    45.1. Should happen in the course page when searching and requesting a course.
46. The system shall give numerical status on the number of semesters left before graduating.
    46.1. Should be shown in both student profile and roadmap.
47. Each course shall have numerical options of perquisites.
    47.1. Should be the default and shown in the course page.
48. Each course shall have an enrollment status based whether a grade has been given or not.
    48.1. Each student should wait until grades are posted to see enrollment status.
49. The system shall not allow students to enroll in courses they do not qualify for.
    49.1. Should be the default in the course page.
50. The system shall require students to complete prerequisites to enroll.
    50.1. Should be the default in the course page.
51. Each course shall display the semester's it is being offered for.
    51.1. Should be the default in the course and roadmap page.
52. Each semester shall display the courses that can be taken prior to enrollment.
    52.1. Should be the default in the roadmap page.

53. The main page shall display the user's schedule and classes with professor’s name, date, and class number.
    53.1. I want to say this should be in the roadmap page.
54. ~~All classes that can be selected will be green while classes that can not be taken will be red. If the course being taken is a prereq the following course will show yellow.~~ Moved to non-functional
55. ~~System will be able to support a large number of users at one time.~~ Moved to non-functional
56. ~~System will run on Windows, Mac and Lynx.~~ Moved to non-functional
57. System will be available during all hours of the day including weekends and holidays.
    57.1. This should be the default.
58. Students shall be authenticated by the university to use the system.
    58.1. This should happen, but since we don’t have access to the SFSU system, it’ll be default.
59. An authenticated person shall use the software.
    59.1. For the register accounts only, and to keep track of unique students.
60. The website shall serve a limited number of people at a time.
    60.1. For enrolled students only, otherwise it’ll cause storage problems.
61. The amount of people shall be defined by the college administration.
    61.1. For only the CS Department for now since we're dealing with a specific department only.

UI Mockups and Storyboards



Title: References for Mockups Page: 2

Web URL: Course/ReqCheck Home Priority:

[C] ReqCheck | Home About Us | Register Login

Degree Planner/Course

Clickable tabs → [ALL] [Future] [In Progress] [Completed]

Every course in progress with info on which fulfillment

General Education	^
A1 - Oral Communication COM 101	L
B2 - Philosophy PHIL-12C	L
Computer Science	^
Intro to Programming CSC 210	L
Lab CSC 211	L

→ In progress symbol

→ correct change

[D]

Web URL: Course/ReqCheck Home Priority:

ReqCheck | Home About Us | Register Login

Degree Planner/Course

clickable tab → [ALL] [Future] [In Progress] [Completed]

shows every completed courses

General Education	20 course	V
A2 - Critical Thinking	3 units	V
C1 - Social Civics	3 units	~
UD-A state History	[Retake] 3 units	X
Computer Science	9 units	^
Mathematics and Physics	12 unit	^

→ displays complete for each course

→ shows warning for courses not passed

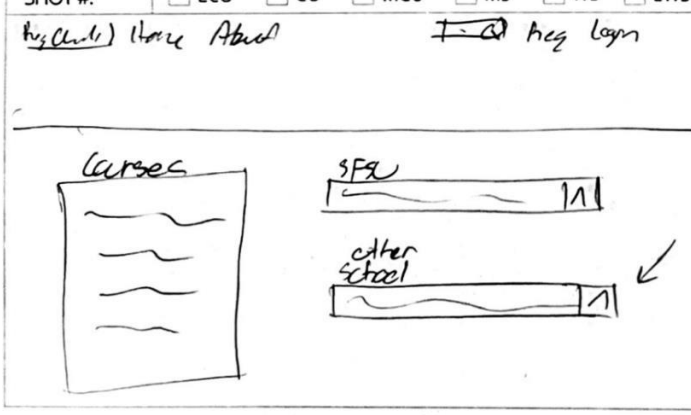
Button to retake courses

PROJECT Course Equivalency Approval UC-02 PAGE ___ of ___

SHOT #: ☐ ECU ☐ CU ☐ MCU ☐ MS ☐ WS ☐ EWS

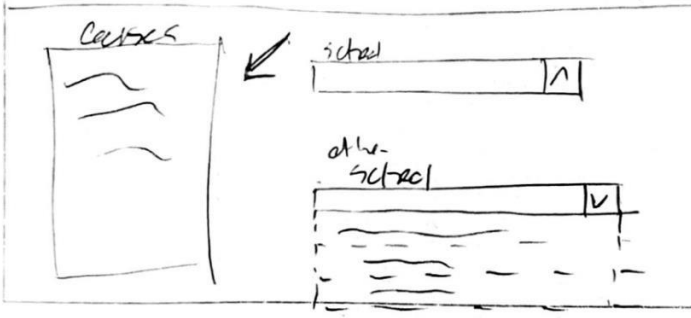
clicks on course equivalency

SHOT #: ☐ ECU ☐ CU ☐ MCU ☐ MS ☐ WS ☐ EWS



clicks on the option to select school's course.

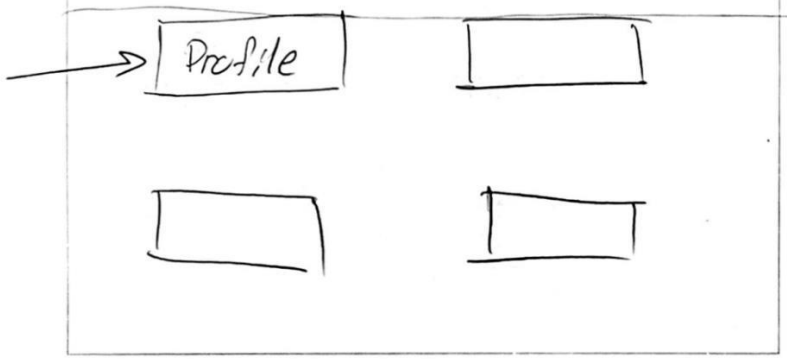
SHOT #: ☐ ECU ☐ CU ☐ MCU ☐ MS ☐ WS ☐ EWS



You see a list of schools and choose a course to see if equivalent.

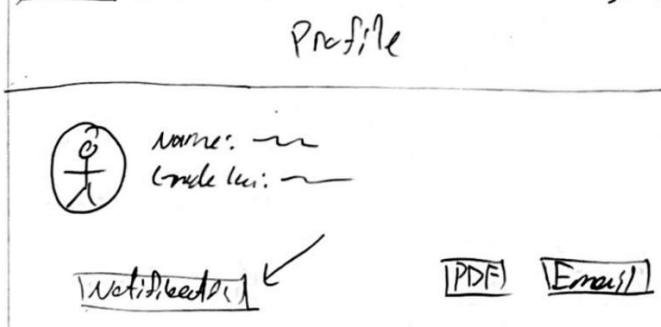
PROJECT Student Exemptions from Department Policy SCENE UC-03 PAGE ___ of ___

SHOT #: ☐ ECU ☐ CU ☐ MCU ☐ MS ☐ WS ☐ EWS



Clicks on profile from profile.

SHOT #: ☐ ECU ☐ CU ☐ MCU ☐ MS ☐ WS ☐ EWS



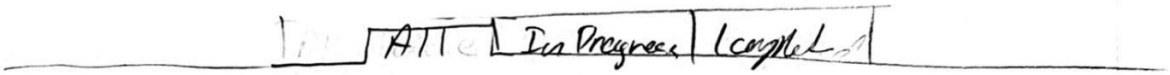
Clicks on notifications

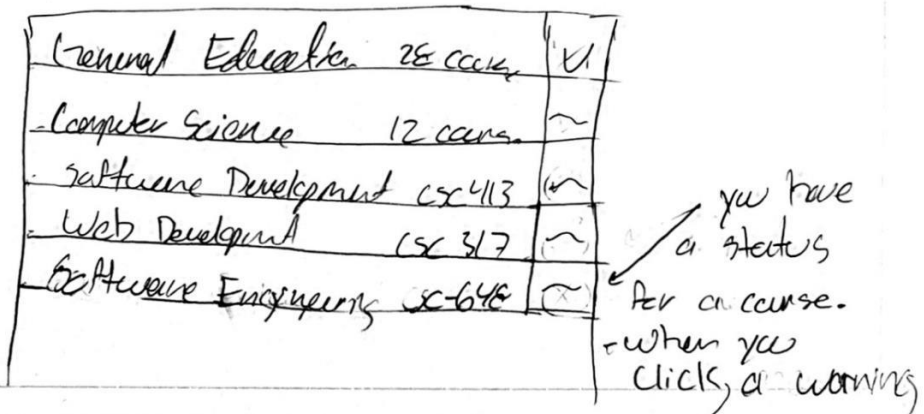
SHOT #: ☐ ECU ☐ CU ☐ MCU ☐ MS ☐ WS ☐ EWS

A pop up shows up with the Exemption Policy

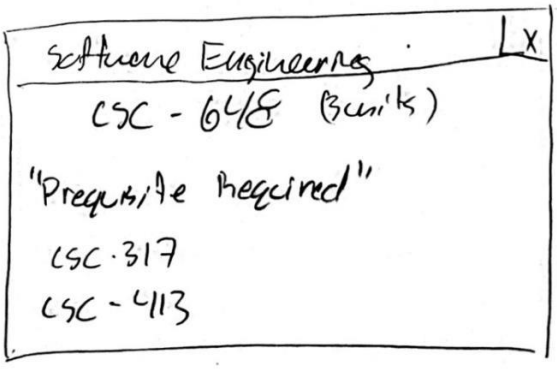
Title: UC-04 | Page:

Web URL: Mockup Reference #3 | Priority:





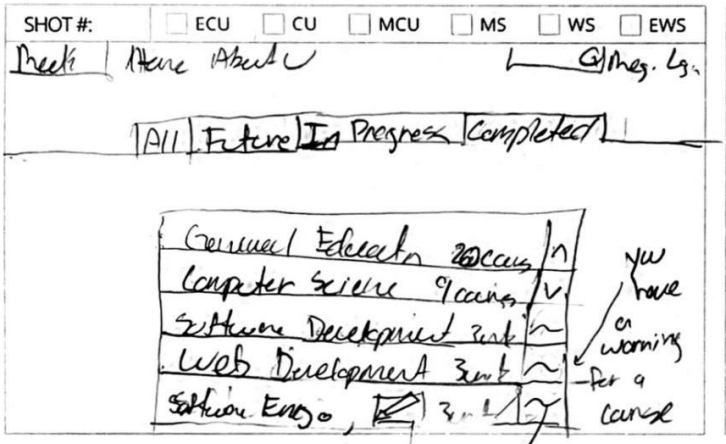
Web URL: Pop up on red cross click | Priority:



you get a pop that says you must fulfill required prequisite

PROJECT UC-05 SCENE PAGE of

SHOT #: ☐ ECU ☐ CU ☐ MCU ☐ MS ☐ WS ☐ EWS

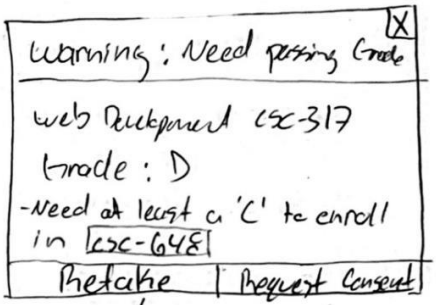


Mockup Reference #13

You have a warning for a taken course.

When you click on the warning

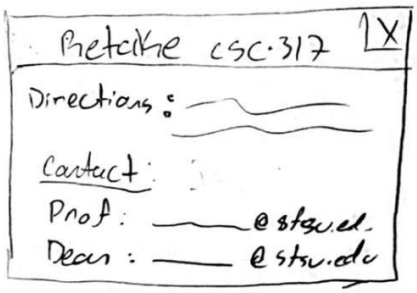
SHOT #: ☐ ECU ☐ CU ☐ MCU ☐ MS ☐ WS ☐ EWS



You get a popup informing you about the warning.

Since you want to retake a course, you click on retake

SHOT #: ☐ ECU ☐ CU ☐ MCU ☐ MS ☐ WS ☐ EWS



You'll receive another pop-up with additional information

Title: UC-09 Page:

Web URL: Profile Priority:

RegChull | Home | About Us | Register | Login

Profile

F-name + L-name

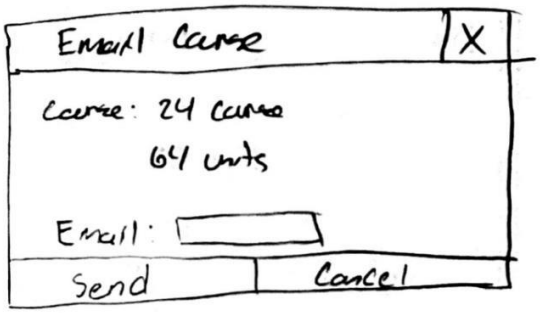
Grade level: Freshman

Expected Graduation: 2026

Notifications:

Course Report: PDF | Email

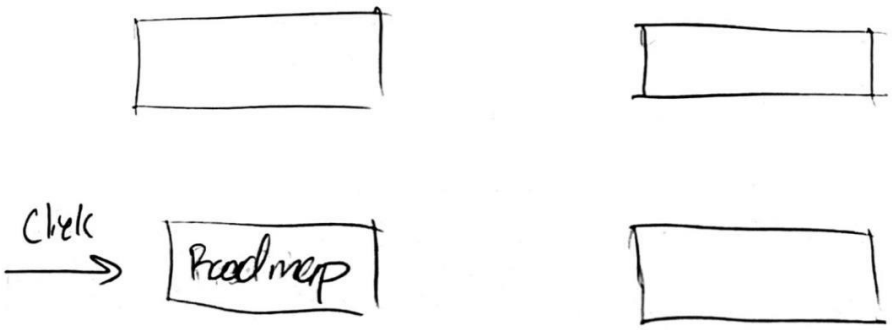
Web URL: Priority:



Title: Prequisite Road Map Chart UC-10 | Page:

Web URL: | Priority:

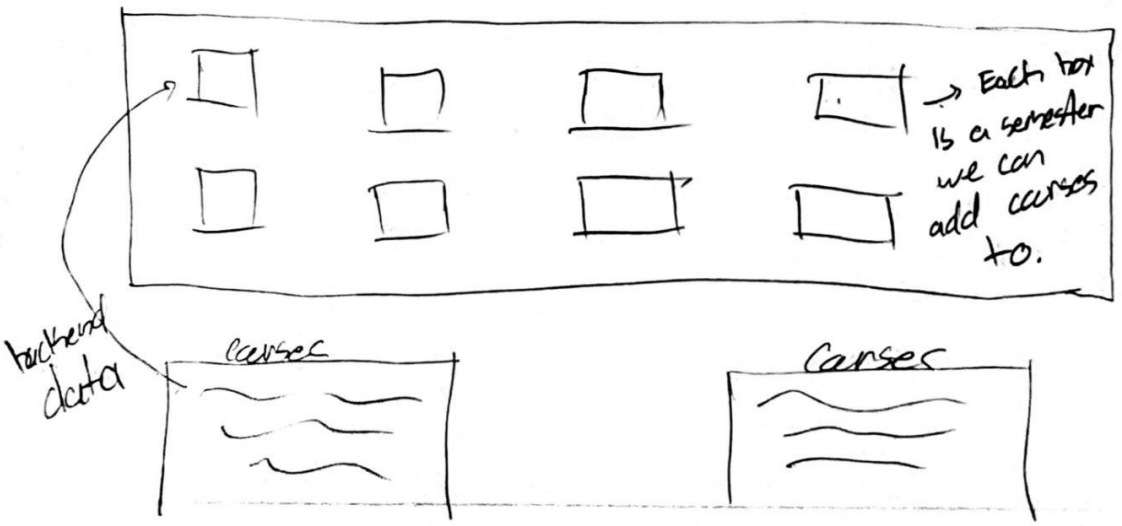
RegCheck | Home | About | Register | Login



Web URL: | Priority:

RegCheck | Home | About | Register | Login

Roadmap

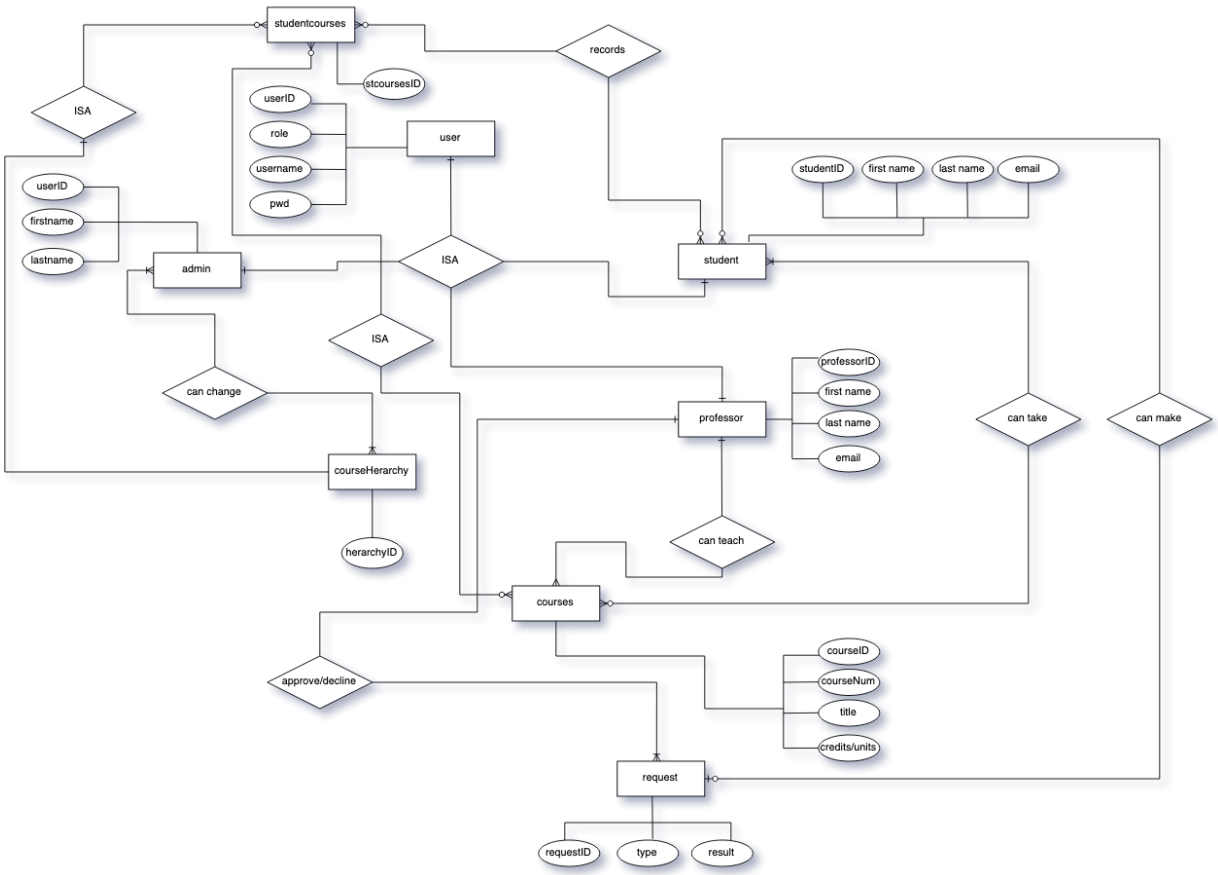


High-level DB Architecture and Organization V2

Database Requirements

1. The admin shall be the user super.
2. The user should have one of these roles: admin, professor or student.
3. All users must have login credentials.
4. Students shall request to get many prerequisites waived.
5. Professor must approve or decline many prerequisite waiver requests.
6. The admin shall update the many course hierarchies for the students.
7. Student courses shall have information about many courses.
8. Students courses shall have be linked to many courses to show one or many prerequisites
9. A professor can teach one or many courses.
10. A student prerequisite request shall be terminated after a defined time.

Entity Relationship Diagram



Entities and Attributes

Entity 1: courses
courseID, primary key and not null.
courseNumber, numeric.
course title, numeric.
credit /units numeric.

Entity 2: admin
empID, primary key and not null.
First name, varchar alphanumeric.
Last name, varchar alphanumeric.

Entity 3: user
userID, primary key and not null.
Role, varchar
Username, alphanumeric and not null
Password, alphanumeric.

Entity 4: student
studentID, primary key, unique, and not null.
First name, alphanumeric.
Last name, alphanumeric.
Email, alphanumeric.

Entity 5: professor
professorID, primary key, unique, and not null.
First name, varchar alphanumeric.
Last name, varchar alphanumeric.
Email, varchar alphanumeric.

Entity 6: student-courses
studentID, fk key
courseID fk key

Entity 7: request
requestID, primary key
Request type, alphanumeric

Result, bool and not null

Entity 8: courseHierarchy
hierarchyID, primary key
courseID, fk key
courseIDs, array

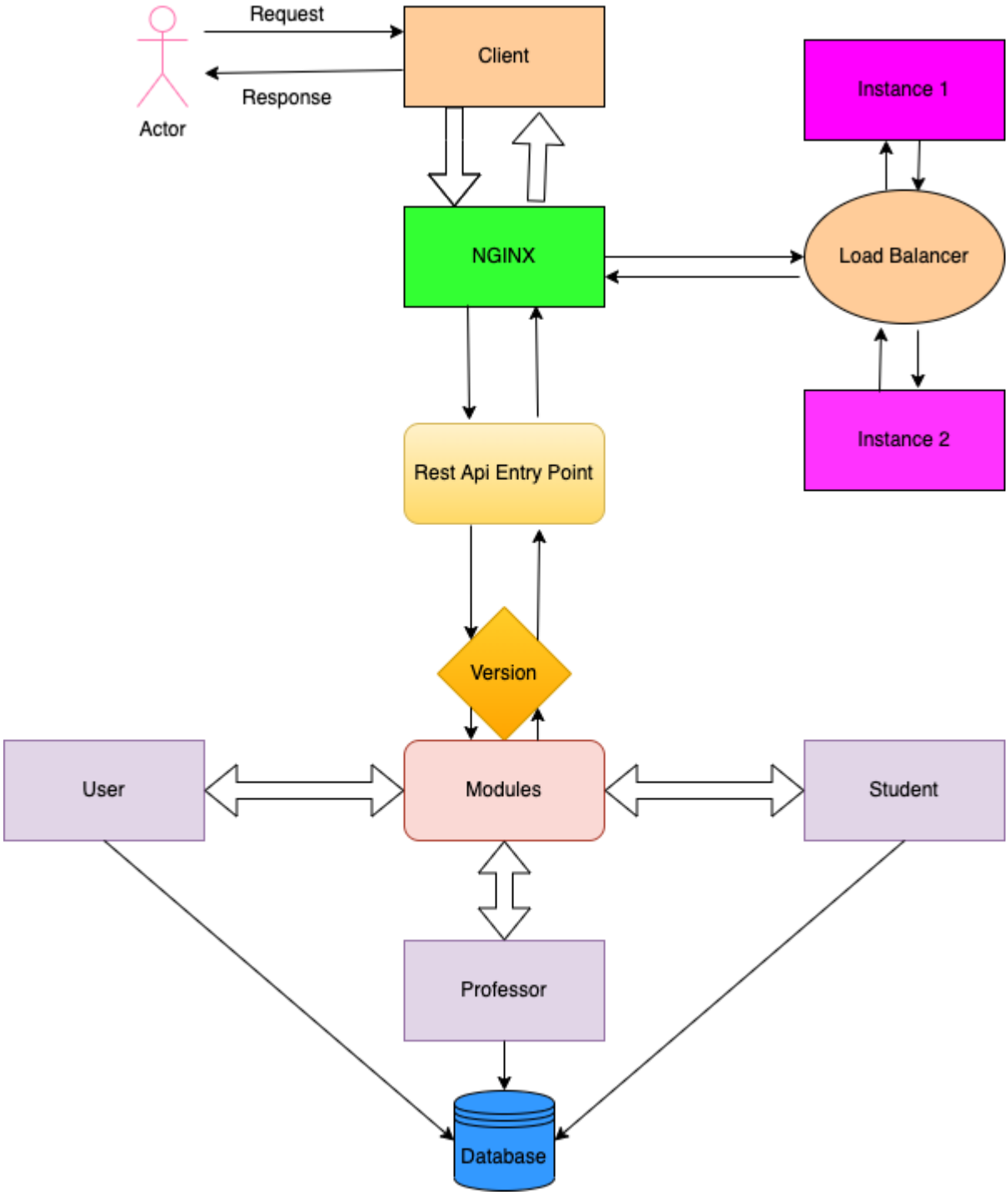
We will use MySQL workbench to manage the backend/database search algorithm. There is already sample data added to the tables so the users can click on the search bar and get recommendations on what to search for. In addition, we will use %s and %i to retrieve data in the workbench to verify backend data.

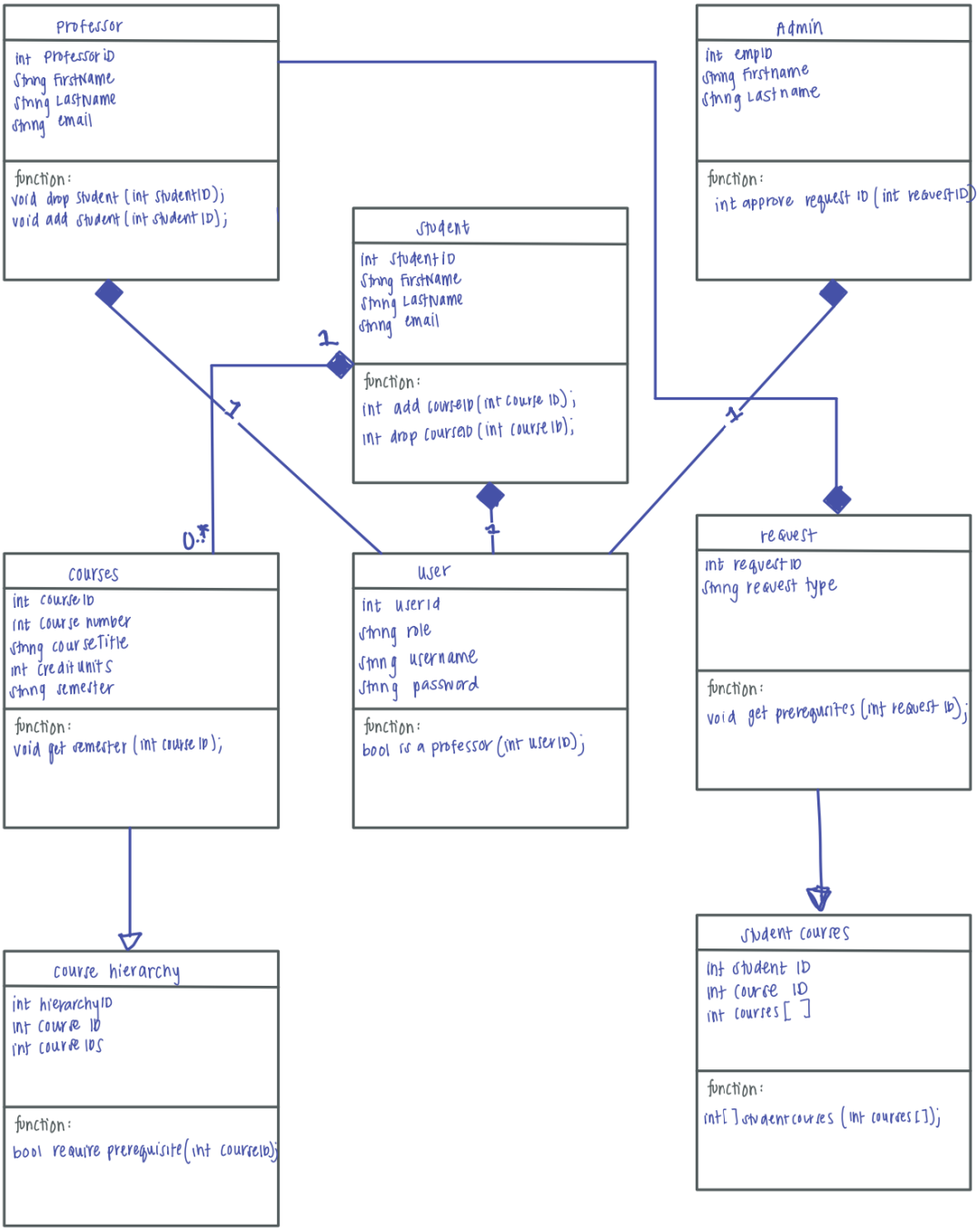
High level API and Main Algorithms V2

Backend Architecture :- **Monolithic architecture**

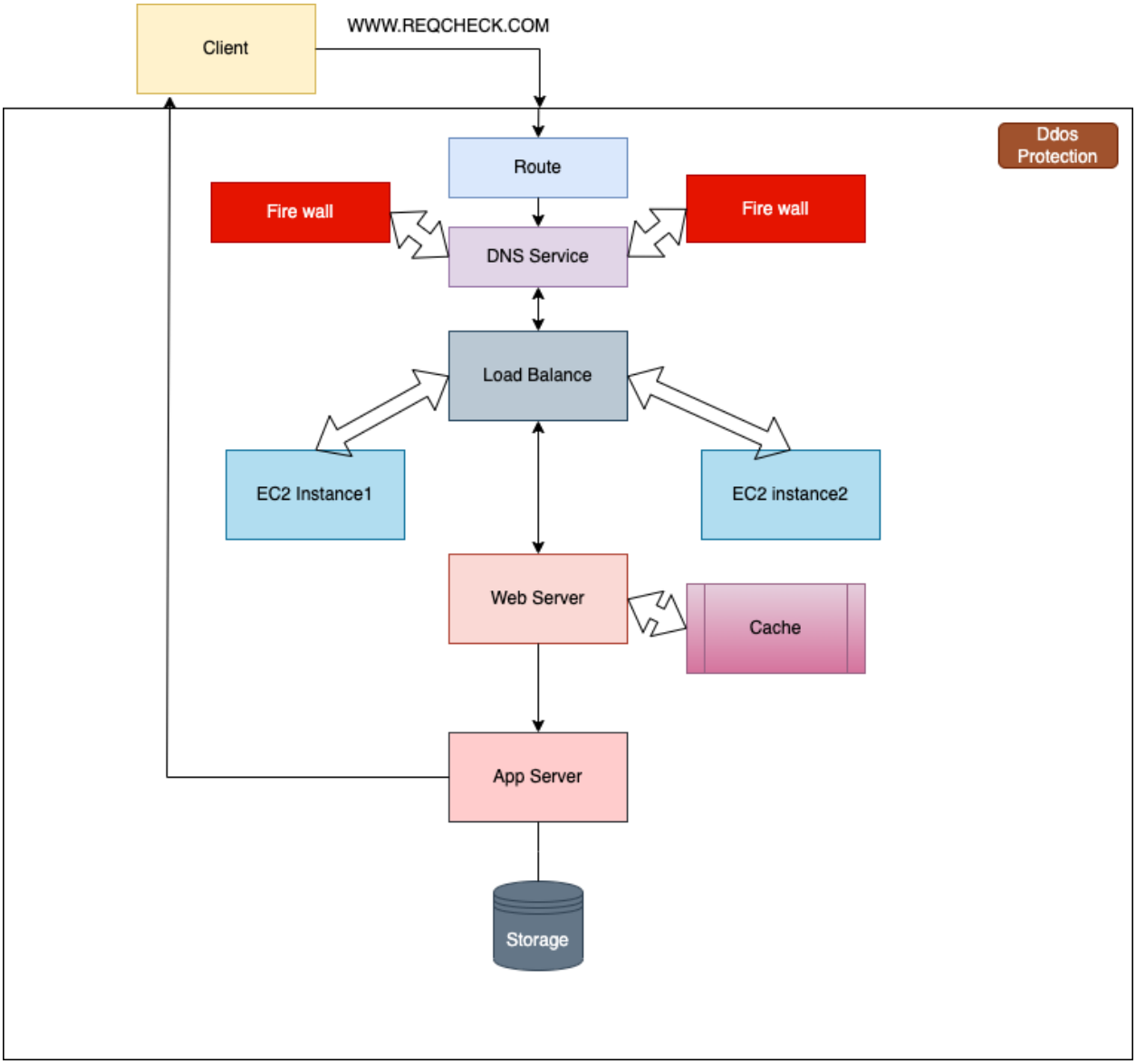
Frontend Architecture:- **Atom architecture**

Algorithms:- **We will mainly use recursive binary search tree to extract the data which are stored as hierarchy.**




Professor
int professorID
String FirstName
String LastName
string email
function:
void drop student (int studentID);
void add student (int student ID);
Admin
int empID
string Firstname
string Last name
function:
int approve request ID (int requestID)
student
int studentID
String FirstName
string LastName
string email
function:
int add courseID (int course ID);
int drop courseID (int courseID);
1
1
1
1
0..*
courses
int courseID
int course number
string courseTitle
int Credit units
string semester
function:
void get semester (int course ID);
User
int userid
string role
string username
string password
function:
bool is a professor (int userID);
request
int request ID
string request type
function:
void get prerequrites (int request ID);
course hierarchy
int hierarchyID
int course ID
int course IDS
function:
bool require prerequisite (int courseID);
student courses
int student ID
int course ID
int courses[ ]
function:
int[ ] studentcourses (int courses[ ]);

High-level Application Network & Deployment Diagram V2



Key Risk V2

- Skills Risk
    - Identify Risk & Explain
        - Most of our team members are exposed to frontend development, but very few in backend development. Not many are fully familiar with React, as well as deployment and cloud platforms. The more technical aspects might need some quick rundown for everyone due to how difficult it is.
    - Solution to issue
        - We'll need to at least get everyone exposed to the full app development; go over the frontend, backend and deployment of an app. We'll probably need to individually go over YouTube links for React and reinforce the knowledge by going over previous milestone prototype code.

- Schedule Risk
    - Identify Risk & Explain
        - Some of the team members will have to make time and allocate the resource due to additional academic courses and non-academic commitments and priorities.
    - Solution to issue
        - We primarily use discord to communicate, and so far, it's working. We just need to communicate more, define our roles/responsibilities, and clarify what needs to be done. Trello would eventually be used as well, and we have a schedule to input our availability.
            - https://www.when2meet.com/?16641975-7JBDt

- Technical Risk
    - Identify Risk & Explain
        - There is not much technical risk to be identified.
    - Solution to issue
        - Any technical risk will be dealt with as soon as possible.

- Teamwork Risk
    - Identify Risk & Explain
        - The primary risk is communication; we have a severe communication issue that is still present and will need to be addressed. We primarily use discord, but I believe we will need to include more face-to-face meetings as well as Zoom meetings. Secondly, our individual roles and responsibilities are not as clear as they should be as well as creating a set timeline to meet deadlines and planning.
    - Solution to issue
        - We'll be implementing changes to how we communicate; meeting over zoom, whether in groups or individuals should be carried out. At least 3x a week if possible and taking notes on said meetings in order to share

with others. We will also need to clarify what our roles and responsibilities will be when we take on a milestone. We'll also need to make aware of what our planning and progress will be like to better understand how the team is doing overall. Also, everyone should support one another by bringing more participation and energy to meetings.

- Legal/Content Risk
    - Identify Risk & Explain
        - There is not much legal risk to be identified since our work is derived from open source.
    - Solution to issue
        - Any legal risk will be dealt with as soon as possible.

Project Management V2

This will be in two parts, before and after the changes in team lead. I would like to acknowledge Vivek's team leadership, so I'll leave his part here.

In order to tackle the work we will divide each module across all the team members. Each team member will have their tasks which are divided into modules. To reduce the dependency, we break the module in such a way that other team member will be not blocked or dependent.To reduce the dependency, we have to give priority to each task.To maintain the task we will use the Trello where all the people will have their ticket which we will includes all details related to the task including attachment.We will sue the development cycle. After completion of each task we will deploy them into the development server.Respective team members will do the unit testing after that task will be reviewed by the team lead and after doing successful testing we will deploy that feature in production. Every team member will finish the high priority task first and so on.We will use Discord platform to communicate with other team member.Every production bug will have the priority to solve that bug.We will have weekly virtual meetings in order to make sure every team member has the idea of overall development.

For project management, we should have used Trello more actively; the usage of DIscord will and did help, but given the time constraints due to leadership, Discord became the primary point of communication. For future tasks, Trello will be used for visual confirmation of individual accomplishments; Discord will be used for group communications in order to verify tasks and ask any questions. Finally, we will use Zoom to meetup and refine and agree to our roles and responsibilities. With Trello, Discord, and Zoom, I hope we can better collaborate and improve our communications.

List of Contributions V2

Part A: Team Lead's evaluation of individual team member's contributions.
Team Lead: Alex Sanchez

Team Member Name	Contributions	Rating
Syed Faiz	Backend Lead	9 / 10
Victoria Wilson-Anumudu	GitHub Master	6 / 10
Vivek Santoki	Front End Lead	7 / 10
Eric Falk	Front end	5 / 10
Erik Rodriguez	Front end	2 / 10

Rating Scale: 1 (No work at all) – 10 (Contributed to the maximum capacity)

- Syed Faiz
    - Contributions
        - Documentation
            - Task Assigned: 4
        - Prototype
            - Setup the AWS database, setup table to connect to frontend when possible.
            - Added an individual "about me" section.
    - Rating: 9 / 10
        - Documentation: 5 / 5
        - Prototype: 4 / 5
            - Implemented backend part for prototype
            - Needs to implement search feature for backend
            - Added an "about me" code to web app

- Victoria Wilson-Anumudu
    - Contributions
        - Documentation
            - Task Assigned: 1, 6
        - Prototype
            - Added an individual "about me" section.
    - Rating: 6 / 10
        - Documentation: 5 / 5
        - Prototype: 1 / 5
            - Added "about me" code to web app

- Vivek Santoki
    - Contributions
        - Documentation
            - Task Assigned: 5, 7
        - Prototype
            - Created the first prototype; full working app, deployed to AWS.
            - Will add an individual “about me” section.
    - Rating: 7 / 10
        - Documentation: 5 / 5
        - Prototype: 2 / 5
            - Implement first prototype; full working app, deployed to AWS, but will not be used.
            - Added an “about me” code to web app

- Eric Falk
    - Contributions
        - Documentation
            - Task Assigned: None
            - Originally all tasks were assigned at the beginning of the milestone, and none were given to Eric.
                - I did not change this due to how close the due dates were when I took over as team lead.
        - Prototype
            - Created the backup prototype; full working app, deployed to both Google Cloud and AWS.
            - Will add an individual “about me” section.
    - Rating: 5 / 10
        - Documentation: 0 / 5
            - No documentation tasks assigned.
        - Prototype: 5 / 5
            - Implement backend prototype; full working app, deployed to AWS & Google Cloud. This prototype will be chosen for submission.

- Erik Rodriguez
    - Contributions
        - Documentation
            - Task Assigned: None
            - Originally all tasks were assigned at the beginning of the milestone, and none were given to Erik.
                - I did not change this due to how close the due dates were when I took over as team lead.
            - I want to note that Erik contributed to task 03, UI Mockups through verbal feedback on UX/UI, but that is about it.
        - Prototype

                - Will add an individual “about me” section.
        - Rating: 2 / 10
            - Documentation: 1 / 5
                - No documentation tasks assigned.
                - Contributed to task 03 verbally.
            - Prototype: 1 / 5
                - Added an individual “about me” section.
- Note
    - I would like to rectify how each team member contributed to both documentation and prototype in the next milestone. At the point I took over as team leader, I erred on not reassigning both documentation and prototype tasks since I prioritize finishing the documentation first before developing the prototype. With time constraints, I prioritize the overall milestone objective over individual contributions. However, this will change and I will make sure every team member will participate, contribute, and understand both documentation and prototype in the next milestone.
    - Given the ratings and notes, I hope my team members will understand my position.

San Francisco State University
SW Engineering CSC 648/848 Fall 2022
Req Check
TEAM #6

Names	**Role**	**Emails**
Alex Sanchez	Team Lead	asanchez26@mail.sfsu.edu
Victoria Wilson-Anumudu	GitHub Master	vwilsonanumudu@mail.sfsu.edu
Syed Faiz	Back End Lead	sfaiz@mail.sfsu.edu
Eric Falk	Front End Lead	efalk1@mail.sfsu.edu
Vivek Santoki	Front End	vsantoki@sfsu.edu
Erik Rodriguez	Front End	erodriguez7@mail.sfsu.edu

Milestone 3

Milestone Version	Date
M3V2	December 1, 2022
M3V1	November 8, 2022

Contents

Data Definitions V3

Entity	Attributes
Student	First name: Varchar (30) Last name: Varchar (30) Student ID: Numeric (10) Email: Varchar (30)
Professor	First name: Varchar (30) Last name: Varchar (30) Employee #: float (10) Email: Varchar (30)
Course	Course Number: Numeric (5) Title: Varchar (45) Units/Credit: Decimal (2) Semester: Varchar (10)
Records	Transcripts: BLOB possibly can change*** Student ID: Numeric (10)
Road Map	Course Number: Numeric (5) Course Title: Varchar (45) Semester: Varchar (10)
Enrollment	Credits: Decimal (2) Course Title: Varchar (45) Student ID: Numeric (10)
Grade	Letter: Varchar (10) Student ID: Numeric (10) Course Number: Numeric (5) Course Title: Varchar (45) Semester: Varchar (10)
Transfer Credits	Institution Name: Varchar (45) Course Number: Numeric (5) Course Title: Varchar (45) Units/Credits: Decimal (2) Student ID: Numeric (10)
Equivalency Appeal	Appeal ID: Pk Key Outcome: Boolean

Notification	Email: Varchar (30) Course Number: Numeric (5)
Attendance	Present: Boolean Absent: Boolean Total Count: SUM numeric (2)
Waitlisted Students	First name: Varchar (30) Last name: Varchar (30) Student ID: Numeric (10) Email: Varchar (30)
Dropped Students	First name: Varchar (30) Last name: Varchar (30) Student ID: Numeric (10) Email: Varchar (30)
Deadlines	Last day to add: Date Time Last day to drop: Date Time Semester: Varchar (10)
Course Hierarchy	Course Number: Numeric (5) Course Title: Varchar (45) Semester: Varchar (10)
Request	Request ID: Pk Type: Varchar (20) Description: Varchar (100) Timestamp: Date Time Outcome: Boolean
Student Courses	Course Number: Numeric (5) Title: Varchar (45) Units/Credit: Decimal (2) Semester: Varchar (10)

Prioritized Functional Requirement V3

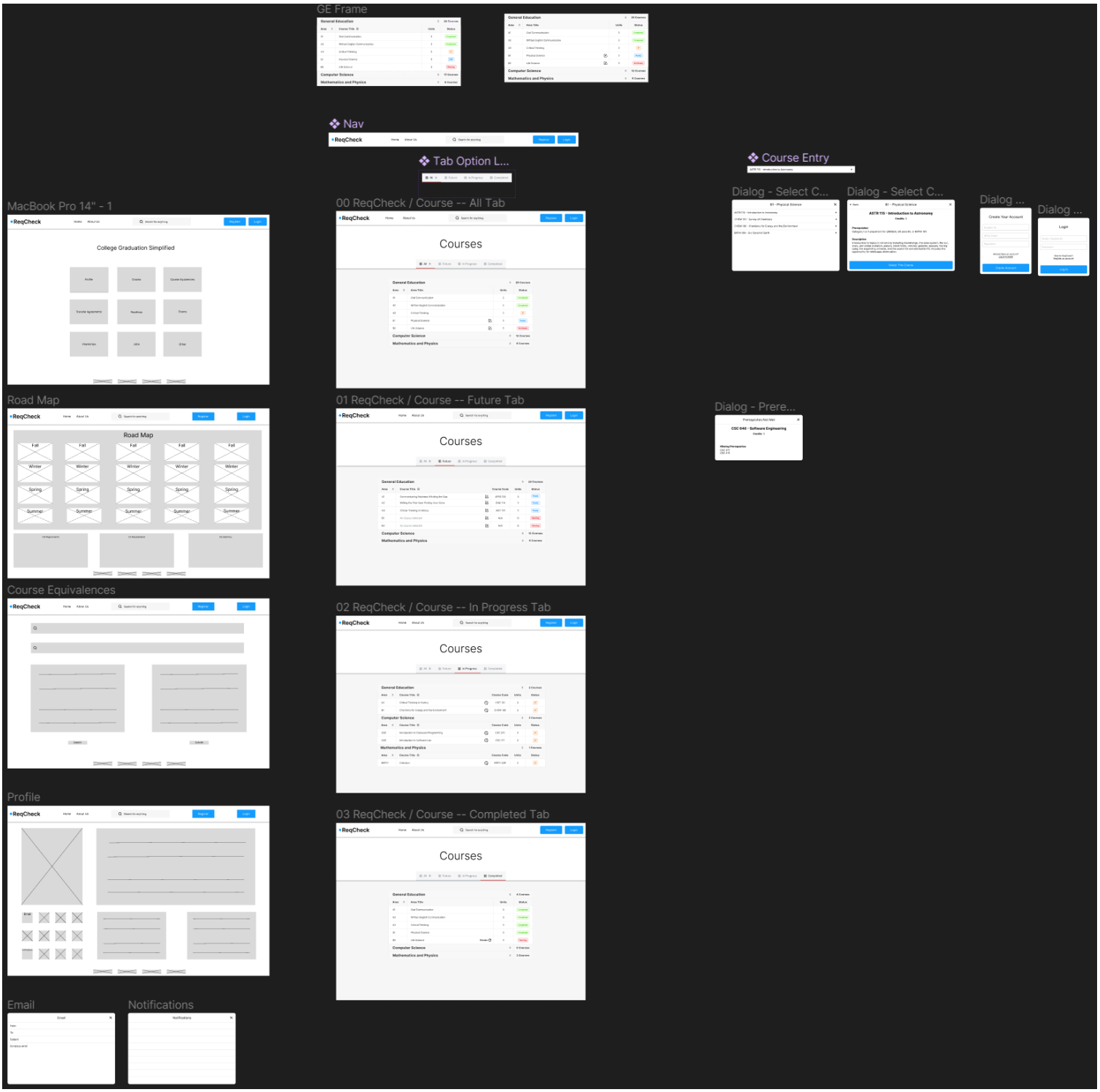
Function	Priority	Actors	Function
4	1	Application	The application should protect the encapsulated user data and disclose information based on roles.
43	1	Course	Each course shall be given a category of major, minor, GE, or elective.
48	1	Course	Each course shall have an enrollment status based whether a grade has been given or not.
1	1	Degree Planner	To use the degree planner feature, students must fill the appropriate subjects that they want to take for their degree.
37	1	Rec Checker	The Req checker shall be available on the WWW platform.
42	1	Requirement	Each requirement change shall update the overall course completion right away.
57	1	System	System will be available during all hours of the day including weekends and holidays.
25	1	System	The system shall display the course requirements for graduation.
26	1	System	The system shall differentiate courses by requirements and electives.
31	1	System	The system shall inform the student any prerequisites required prior to enrolling in a course.
46	1	System	The system shall give numerical status on the number of semesters left before graduating.
3	1	Timeline	The timeline shall depend on how much units a particular student must complete to obtain their degree.
18	1	Student	Students shall have a selection of departments they must choose from.
17	1	Admin	Admin shall add the prerequisite on a defined hierarchy of courses.
20	1	Admin	Admin shall have a list of all the courses.
21	1	Admin	Admin shall have a list of courses taken by a student.
6	2	Application	The application should automatically verify a student's academic records via the administrator.
7	2	Application	The application should be able to authenticate user identity by the university.
51	2	Course	Each course shall display the semester’s it is being offered for.
52	2	Semester	Each semester shall display the courses that can be taken prior to enrollment.
58	2	Student	Students shall be authenticated by the university to use the system.
28	2	System	The system shall update all courses based on students input data.
29	2	System	The system shall inform students of course exemptions.
32	2	System	The system shall update any prerequisite requirements.
34	2	System	The system shall enforce any prerequisites prior to enrolling in courses.
35	2	System	The system shall require satisfaction of prerequisites to enroll in courses.
36	2	System	The system shall give alternatives to satisfying prerequisites if applicable.

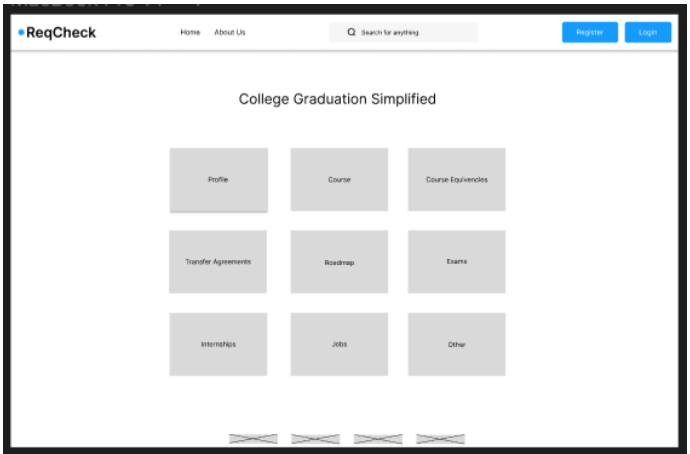
49	2	System	The system shall not allow students to enroll in courses they do not qualify for.
50	2	System	The system shall require students to complete prerequisites to enroll.
30	2	System	The system shall inform the student the number of semesters left to graduate.
14	2	Admin	Admin shall set new records and shall make a hierarchy of prerequisite courses.
15	2	Admin	Admin shall arrange all the courses department wise.
19	2	Course	Course shall have less dependency to any other entity.
45	2	Exemption	Each exemption shall be based on course requirements and student grade level.
53	2	Main Page	The main page shall display the user's schedule and classes with professor’s name, date, and class number.
12	2	Professor	A professor shall fill up a seat once a student had dropped the course.
10	2	Professor	Professors shall only approve a defined number of students to their courses.
11	2	Professor	A professor shall fill in dropped students’ seats using a waitlist of students.
22	2	Student	Students shall see the list of professors for a course.
13	2	Student	Each student shall have a chance to request enrollment and each enrollment seat shall be filled up by a first come first serve basis.
33	2	System	The system shall be notified if a course has not been passed.
23	2	System	The system shall display the student’s grade level (current, transfer, new).
24	2	System	The system shall inform students of any changes in department requirements.
41	2	System	The system shall give a grade level status based on overall course completion.
27	2	System	The system shall notify students of all newly required courses.
47	3	Course	Each course shall have numerical options of perquisites.
44	3	Course	Each course shall be appointed based on a student’s grade level (current, transfer, new).
8	3	System	System shall provide all information on passed courses, professors, grades, and sessions.
16	3	Admin	Admin shall change any courses/prerequisites as per the demand of the admin panel.
61	3	People	The amount of people shall be defined by the college administration.
5	3	User	The user (student) must be given information on whether a specific course is transferable from a certain school.
9	3	System	System shall send notifications of recommended courses and waitlisted courses for users to add.
59	0	N/A	
2	0	N/A	
60	0	N/A	
38	0	N/A	
39	0	N/A	

40	0	N/A	
54	0	N/A	
55	0	N/A	
56	0	N/A	

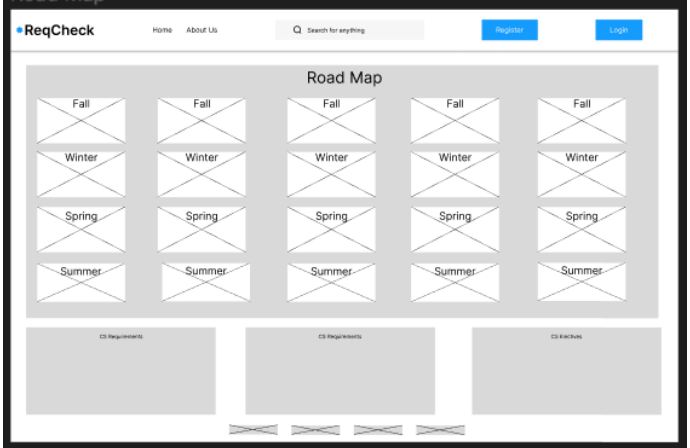
Wireframes based on Mockups and Storyboards

https://www.figma.com/file/NgvaVTZ5rMJ4zJlmch3yOS/Courses-Page-(Copy)

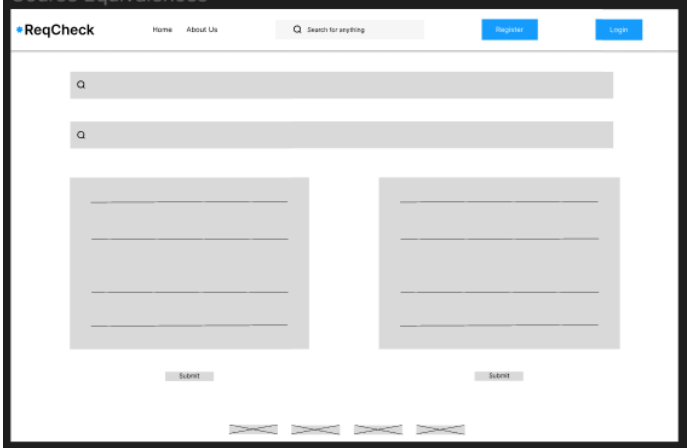




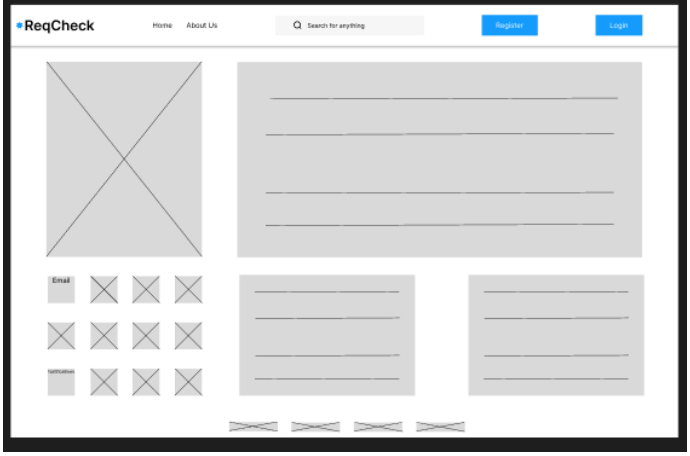
Road Map

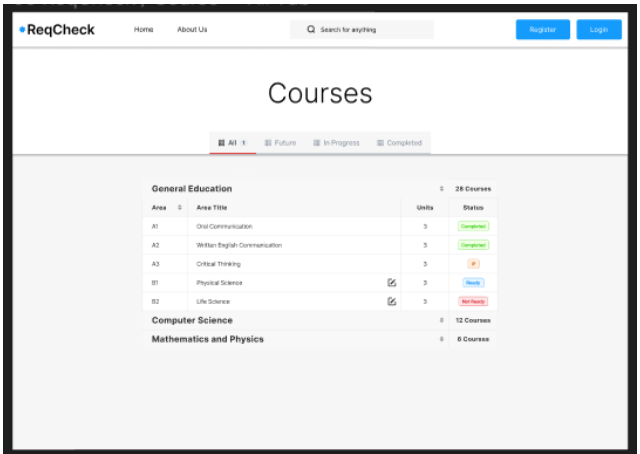


Course Equivalences

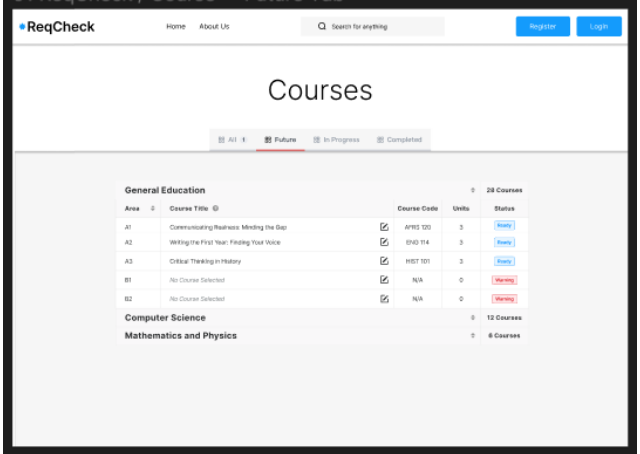


Profile

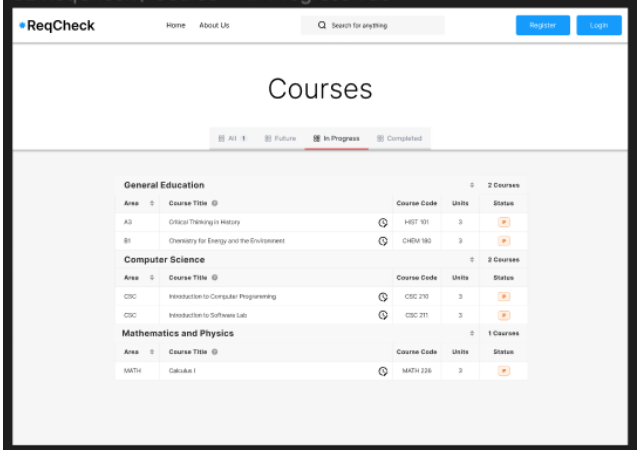




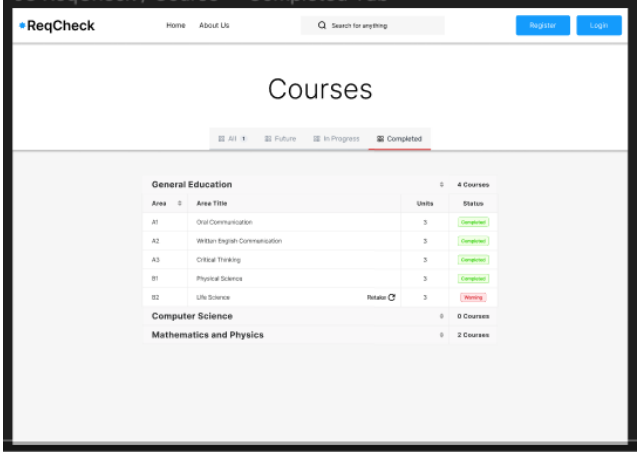
01 ReqCheck / Course -- Future Tab

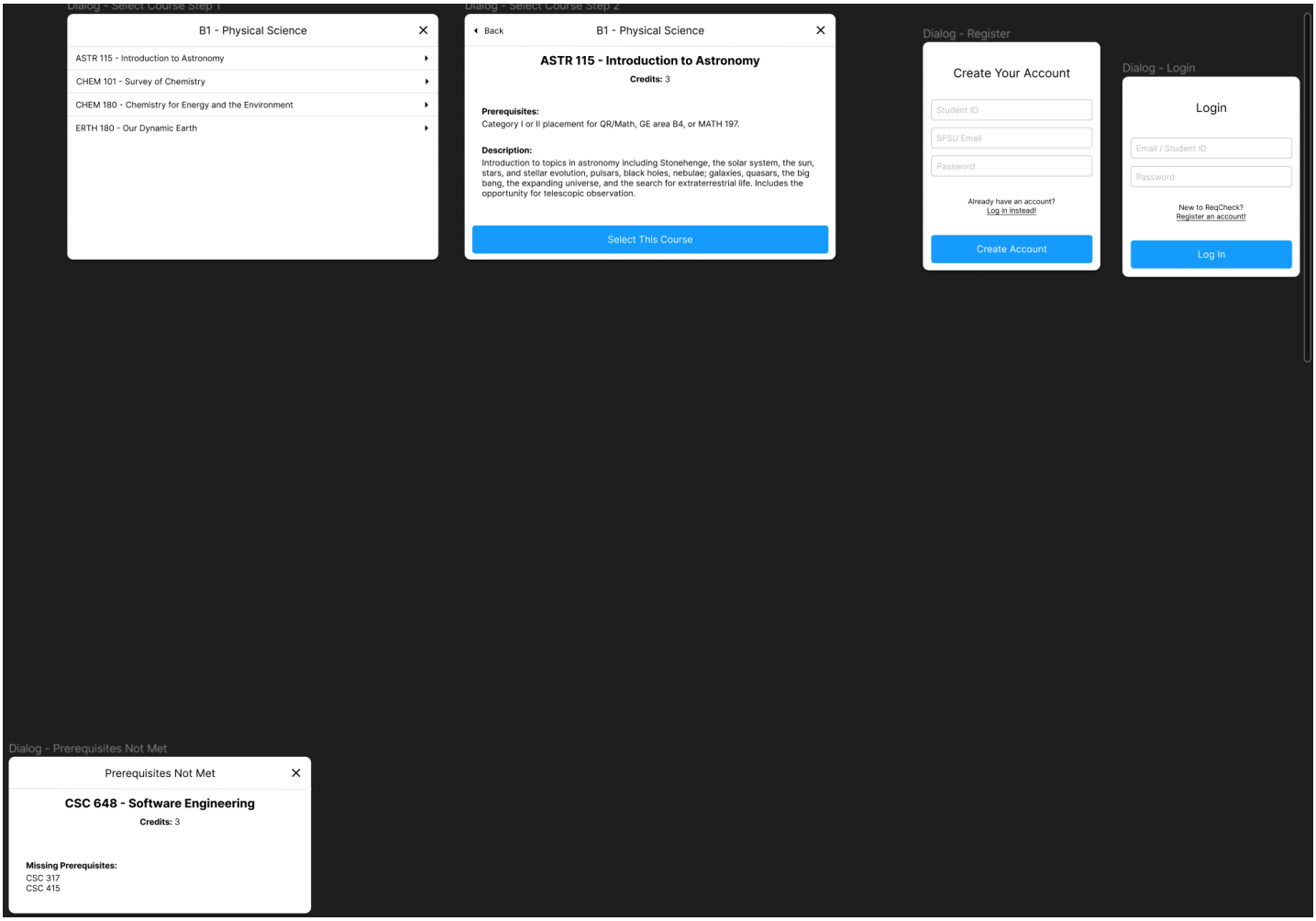


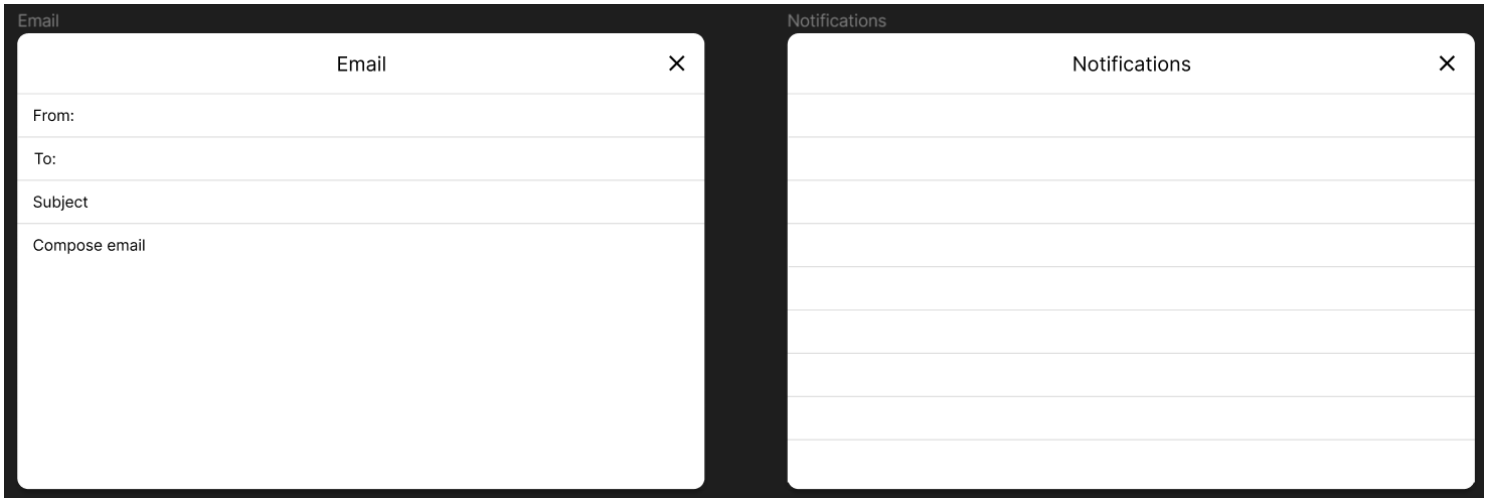
02 ReqCheck / Course -- In Progress Tab



03 ReqCheck / Course -- Completed Tab

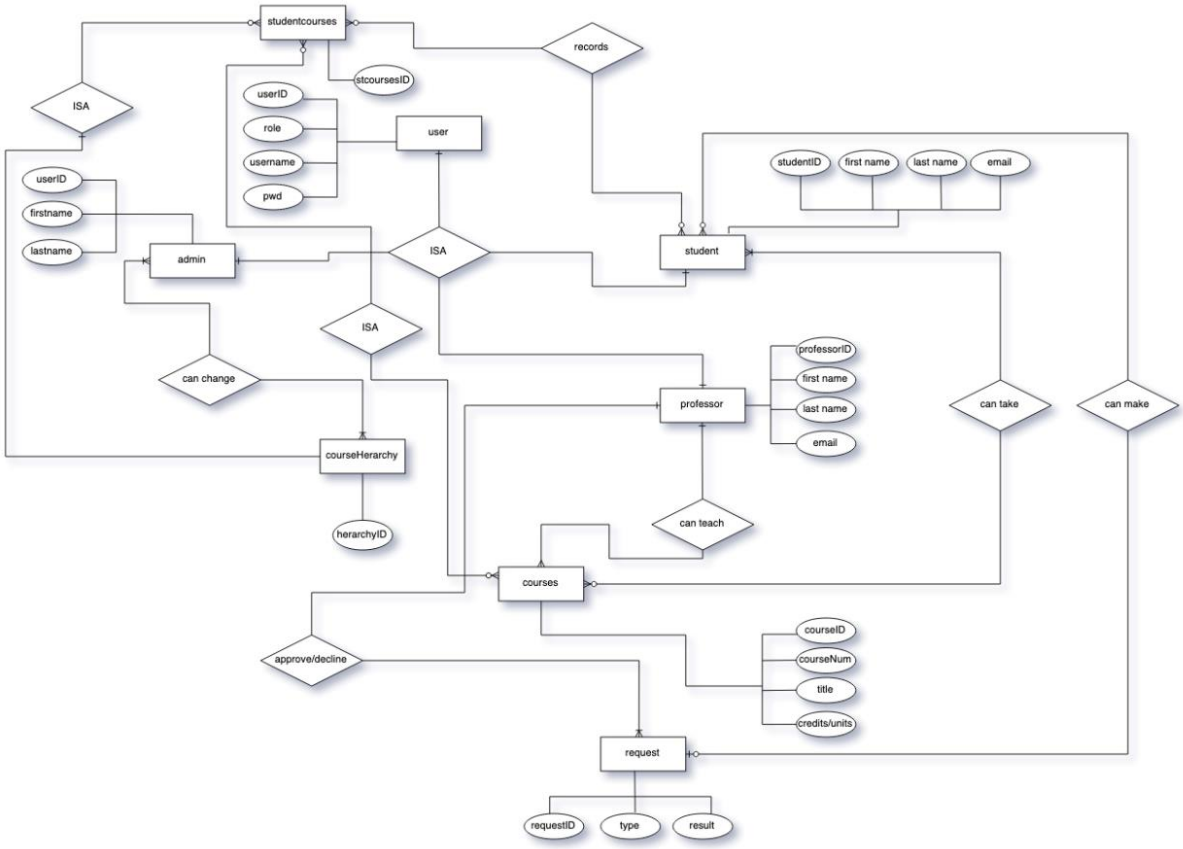



Dialog - Select Course Step 1
B1 - Physical Science
ASTR 115 - Introduction to Astronomy
CHEM 101 - Survey of Chemistry
CHEM 180 - Chemistry for Energy and the Environment
ERTH 180 - Our Dynamic Earth
Dialog - Select Course Step 2
Back
B1 - Physical Science
ASTR 115 - Introduction to Astronomy
Credits: 3
Prerequisites:
Category I or II placement for QR/Math, GE area B4, or MATH 197.
Description:
Introduction to topics in astronomy including Stonehenge, the solar system, the sun, stars, and stellar evolution, pulsars, black holes, nebulae; galaxies, quasars, the big bang, the expanding universe, and the search for extraterrestrial life. Includes the opportunity for telescopic observation.
Select This Course
Dialog - Register
Create Your Account
Student ID
SFSU Email
Password
Already have an account?
Log in instead!
Create Account
Dialog - Login
Login
Email / Student ID
Password
New to ReqCheck?
Register an account!
Log In
Dialog - Prerequisites Not Met
Prerequisites Not Met
CSC 648 - Software Engineering
Credits: 3
Missing Prerequisites:
CSC 317
CSC 415

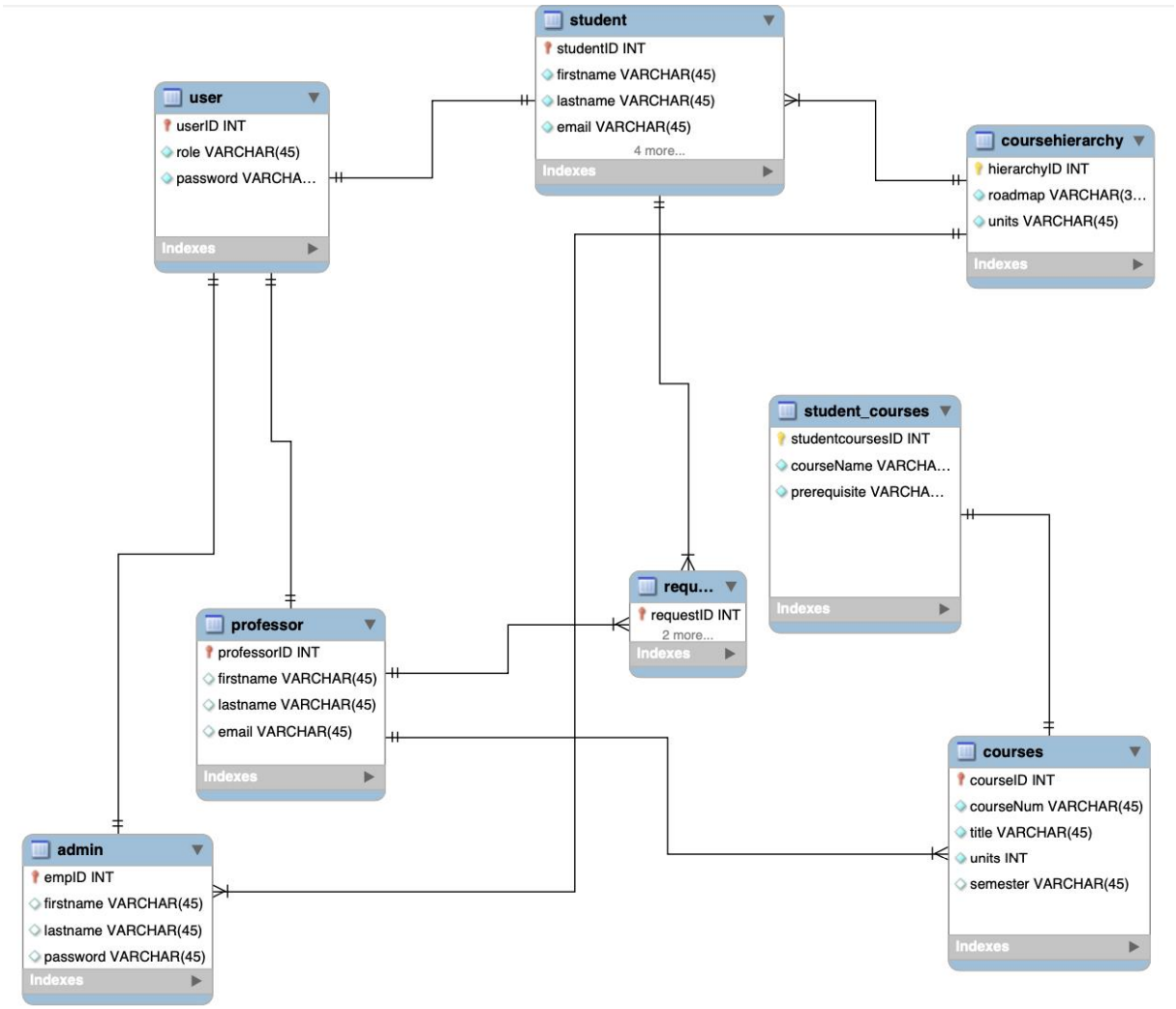

Email
Email
From:
To:
Subject
Compose email
Notifications
Notifications

High-level DB Architecture and Organization V2

Entity Relation Diagram



Enhanced Entity Relation (EER) Diagram



Entities and Attributes

Entity 1: courses
courseID, primary key.
courseNumber, varchar.
title, varchar.
Units, int.
Semester, varchar.

Entity 2: admin
empID, primary key.
First name, varchar.
Last name, varchar.
Password, varchar.

Entity 3: user
userID, primary key.
Role, varchar.
Password, varchar.

Entity 4: student
studentID, primary key, unique.
First name, varchar.
Last name, varchar.
Email, varchar.
Level, varchar.
Units completed, int.
Courses taken, varchar.
Grade, varchar.

Entity 5: professor
professorID, primary key, unique, and not null.
First name, varchar alphanumeric.
Last name, varchar alphanumeric.
Email, varchar alphanumeric.

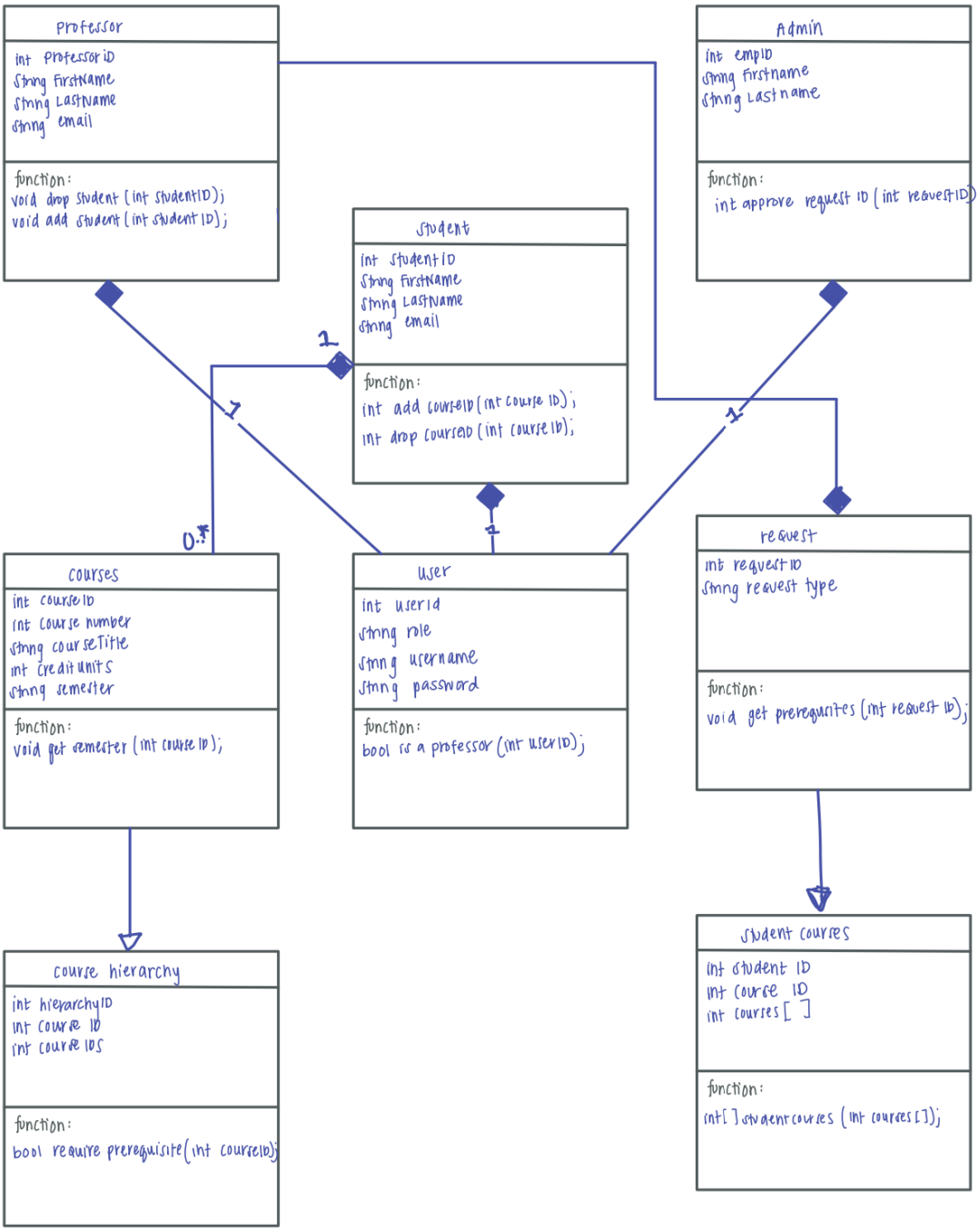
Entity 6: student-courses
student_coursesID, primary key.

Course name, varchar.
Prerequisite, varchar.

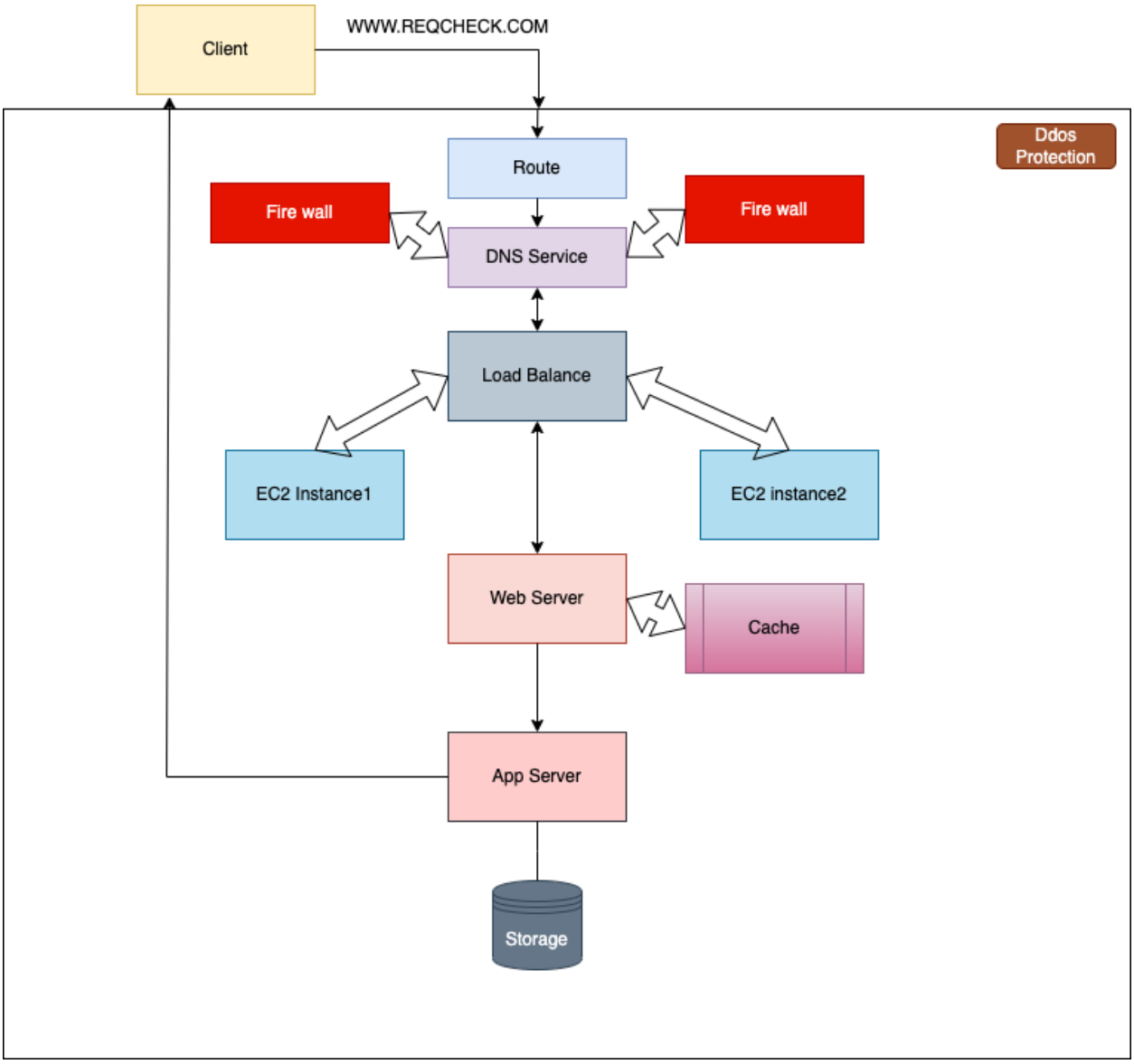
Entity 7: request
requestID, primary key
type, varchar.
outcome, bool.

Entity 8: courseHierarchy
hierarchyID, primary key
Roadmap, varchar.
Units varchar.
We will use MySQL workbench to manage the backend/database search algorithm. The database already has tables with sample data to help users get recommendations for search when they click on the search bar. In addition, we will use %s and %i to retrieve data in the workbench to verify backend data. Furthermore, there will not be any media such as images or videos in the database hence we will not be using blob to avoid memory storage problems.

High Level Diagram V2



High Level Application Network & Deployment Diagram



List of Contributions

To be filled by team lead only:

Team Member Name	Role	Rating
Syed Faiz	Backend Lead	9/10
Victoria Wilson-Anumudu	Github Master	7/10
Eric Falk	Frontend Lead	10/10
Vivek Santoki	Frontend	6/10
Erik Rodriguez	Frontend	7/10

- Syed Faiz
  - Documentation
    - Task Assigned: 1,2, 4
  - Prototype
    - Maintain backend database, confirmation of insertion data from vertical prototype.
  - Rating: 9/10
    - Good communication, always giving feedback and can depend on to carry out task when assigned. Sometimes needs a reminder or two for some rework though.
- Victoria Wilson-Anumudu
  - Documentation
    - Task Assigned:1, 2, 5
  - Prototype
    - No coding, but had review m2v2 usecase's, and m3 Figma work is consistent with the horizontal prototype.
  - Rating: 7/10
    - Consistent work and can depend on doing assigned task. Though communication and participation needs to be improved going forward.
- Eric Falk
  - Documentation
    - Task Assigned: Task 3
  - Prototype
    - Maintain and coded the majority of Team 06 webapp; full deployment, and maintenance of the code as well as implementing task 03 Figma mockups.
  - Rating: 10/10

        - Great communication: consistent with schedule and timeline, always giving feedback, can depend on to carry out work. No criticism. Promoted to Frontend lead.
- Vivek Santoki
    - Documentation
        - Task Assigned: 5
    - Prototype
        - Implemented the vertical prototype coding with confirmation of backend insertion. No coding in horizontal prototype.
    - Rating: 6/10
        - Not consistent with a schedule, will always need to be reminded constantly to carry out assigned task, will not double check work to confirm objectives, uncertain if able to complete task if given. Uncertain in his role and responsibilities.
- Erik Rodriguez
    - Documentation
        - Task Assigned: 2, 3
    - Prototype
        - No coding done.
    - Rating: 7/10
        - Consistent with work and communication. No coding work on horizontal prototype. Collaborated with EricF on horizontal prototype due to coding experience. Will need to step up on coding next milestone.

San Francisco State University
SW Engineering CSC 648/848 Fall 2022
Req Check
TEAM #6

Names	**Role**	**Emails**
Alex Sanchez	Team Lead	asanchez26@mail.sfsu.edu
Victoria Wilson-Anumudu	GitHub Master	vwilsonanumudu@mail.sfsu.edu
Syed Faiz	Back End Lead	sfaiz@mail.sfsu.edu
Eric Falk	Front End Lead	efalk1@mail.sfsu.edu
Vivek Santoki	Front End	vsantoki@sfsu.edu
Erik Rodriguez	Front End	erodriguez7@mail.sfsu.edu

Milestone 4

Milestone Version	Date
M4V2	December 15, 2022
M4V1	December 01, 2022

Table of Contents

Product Summary

Function	Priority	Actors	Function
4	1	Application	The application should protect the encapsulated user data and disclose information based on roles.
43	1	Course	Each course shall be given a category of major, minor, GE, or elective.
1	1	Degree Planner	To use the degree planner feature, students must fill the appropriate subjects that they want to take for their degree.
37	1	Rec Checker	The Req checker shall be available on the WWW platform.
57	1	System	System will be available during all hours of the day including weekends and holidays.
25	1	System	The system shall display the course requirements for graduation.
26	1	System	The system shall differentiate courses by requirements and electives.
5	1	User	The user (student) must be given information on whether a specific course is transferable from a certain school.
19	1	Course	Course shall have less dependency to any other entity.
18	1	Student	Students shall have a selection of departments they must choose from.
31	2	System	The system shall inform the student any prerequisites required prior to enrolling in a course.
46	2	System	The system shall give numerical status on the number of semesters left before graduating.
3	2	Timeline	The timeline shall depend on how much units a particular student must complete to obtain their degree.
48	2	Course	Each course shall have an enrollment status based whether a grade has been given or not.
42	2	Requirement	Each requirement change shall update the overall course completion right away.
17	2	Admin	Admin shall add the prerequisite on a defined hierarchy of courses.
20	2	Admin	Admin shall have a list of all the courses.
21	2	Admin	Admin shall have a list of courses taken by a student.
6	2	Application	The application should automatically verify a student's academic records via the administrator.
7	2	Application	The application should be able to authenticate user identity by the university.
51	2	Course	Each course shall display the semester’s it is being offered for.
52	2	Semester	Each semester shall display the courses that can be taken prior to enrollment.
58	2	Student	Students shall be authenticated by the university to use the system.

28	2	System	The system shall update all courses based on students input data.
29	2	System	The system shall inform students of course exemptions.
32	2	System	The system shall update any prerequisite requirements.
34	2	System	The system shall enforce any prerequisites prior to enrolling in courses.
35	2	System	The system shall require satisfaction of prerequisites to enroll in courses.
36	2	System	The system shall give alternatives to satisfying prerequisites if applicable.
49	2	System	The system shall not allow students to enroll in courses they do not qualify for.
50	2	System	The system shall require students to complete prerequisites to enroll.
30	2	System	The system shall inform the student the number of semesters left to graduate.
14	2	Admin	Admin shall set new records and shall make a hierarchy of prerequisite courses.
15	2	Admin	Admin shall arrange all the courses department wise.
45	2	Exemption	Each exemption shall be based on course requirements and student grade level.
53	2	Main Page	The main page shall display the user's schedule and classes with professor’s name, date, and class number.
12	2	Professor	A professor shall fill up a seat once a student had dropped the course.
10	2	Professor	Professors shall only approve a defined number of students to their courses.
11	2	Professor	A professor shall fill in dropped students’ seats using a waitlist of students.
22	2	Student	Students shall see the list of professors for a course.
13	2	Student	Each student shall have a chance to request enrollment and each enrollment seat shall be filled up by a first come first serve basis.
33	2	System	The system shall be notified if a course has not been passed.
23	2	System	The system shall display the student’s grade level (current, transfer, new).
24	2	System	The system shall inform students of any changes in department requirements.
41	2	System	The system shall give a grade level status based on overall course completion.
27	2	System	The system shall notify students of all newly required courses.
47	3	Course	Each course shall have numerical options of perquisites.
44	3	Course	Each course shall be appointed based on a student’s grade level (current, transfer, new).

8	3	System	System shall provide all information on passed courses, professors, grades, and sessions.
16	3	Admin	Admin shall change any courses/prerequisites as per the demand of the admin panel.
61	3	People	The amount of people shall be defined by the college administration.
9	3	System	System shall send notifications of recommended courses and waitlisted courses for users to add.
59	0	N/A	
2	0	N/A	
60	0	N/A	
38	0	N/A	
39	0	N/A	
40	0	N/A	
54	0	N/A	
55	0	N/A	
56	0	N/A	

Title: ReqCheck

Since most colleges have their own course curriculums for their programs, students find it quite difficult to make these transfers due to discrepancies between their courses. It typically takes about 2 to 3 weeks to finalize/approve a transfer due to demand and a lack of cohesivity between school programs and course codes. Therefore we built this software Req Check, where we typically solve this problem by strategically creating value to save time and provide knowledge to both students and school administrators on how to handle transfers.

Uniqueness

The Req Check software takes one school's course code and matches it with the other schools' courses. An automated system like Req Check will efficiently map out when a student will graduate as well as ensure they are on the right track as a new transfer student. Req Check will alleviate stress for students and faculty as well as save time and money. At the end of the day, the goal is to help both parties and stakeholders plan and handle the transfers beforehand seamlessly.

URL to your product accessible to instructor, on deployment server:

http://ec2-52-53-211-193.us-west-1.compute.amazonaws.com/

Usability Test Plan

Task 1

Objectives

- We will be testing the ability of a user to plan their courses according to their degree requirements. The user should have an easy way to see exactly what requirements are needed for their degree, and they should be able to quickly switch out a course if there are multiple that satisfy a requirement. No matter what change is made on the courses page, it should always end up in a state that represents a valid degree plan. The page should be intuitive for the user, and it should show plenty of information without being overwhelming. The dialogs should be easy to navigate and leave the user with a clear indication of what they are doing.

Description	Change an editable degree requirement
State	The user must be logged in to be able to see the courses page. If not logged in, the user will be redirected to the home page.
Completion Criteria	A course title appears in place of “No Course Selected”
Intended Users	The courses page should only be accessible to students with registered accounts. This specific feature is useful for students who have degree requirements that can be satisfied by multiple courses.
URL	http://ec2-52-53-211-193.us-west-1.compute.amazonaws.com/courses

Task 2

Objectives

- We will be testing the ability of a user to search for any SFSU undergraduate course. The user should be able to easily search a course to view all relevant information about that course. The search bar should be responsive and never leave the user without feedback. When an ambiguous search is made, the user should be presented with a results page displaying all potential matches to the search.

Description	Search for a course at SFSU
State	User is on any page with the search bar accessible
Completion Criteria	The page displays information about a specific course
Intended Users	Any user who want to know more about a specific course
URL	http://ec2-52-53-211-193.us-west-1.compute.amazonaws.com

Task 3

Objectives

- For this function, a registered user will be able to access the roadmap webpage and be able to set up their course roadmap. How this will work is that the registered user will have already compiled their courses and will be able to list them in the semesters they wish to take in the future. This function will allow the registered user to visually see a roadmap of the courses they can take in the future, semester by semester.

Description	Use the roadmap to plan courses by semester
State	A student user should be logged in and on the roadmap page
Completion Criteria	Roadmap displays all courses organized into semesters
Intended Users	Students trying to plan their courses by semester. This page will not be accessible to non-registered users.
URL	http://ec2-52-53-211-193.us-west-1.compute.amazonaws.com/roadmap

Task 4

Objectives

- For this function, we will be testing to see if there are any course prerequisites to a chosen course. When a registered user is on the course webpage, they will be able to select courses and be notified of whether a course has any prerequisites. This function is purely informative and visual. For example, when a registered user is selecting their courses, and they select csc413, they will be advised that the course requires csc340.

Description	Check prerequisites for an upcoming course
State	For the system setup of the state, the user must be a registered user. If the user is not registered, then they will not see this webpage. If a user is registered, they will see the course webpage; they will see all the course requirements as well as whether a course is in progress or completed. Furthermore, each course section requirement sill allow the user to be given information on whether a course has prerequisites. This way, they will be notified if an upcoming course will require prerequisites.
Completion Criteria	When a registered user has chosen a course, they will be notified of any prerequisites. A pop up notification will be both informative and visual in order to emphasize the prerequisite requirement.
Intended Users	This will be only for registered users; a non-register user will not see this option.
URL	http://ec2-52-53-211-193.us-west-1.compute.amazonaws.com/courses

Task 5

Objectives

- For this function, we will be testing to see if courses from another school are equivalent to a course at San Francisco State University. If they are equivalent, then you will be given credit for the course taken at the other school. The reason why is if you are a transfer student, you should have taken previous courses that will have allowed you to transfer to SFSU in order to receive course credit for the Bachelor's Degree of Computer Science. This function will thus allow you to test whether a certain course is equivalent to a course at SFSU.

Description	Check a course equivalency; will test the equivalency of a course taken at another school to a current course at San Francisco State University.
State	For the system setup of the state, any user can check in to see the course equivalency. You can choose the school, and then choose the course applicable.
Completion Criteria	After choosing the courses, you will be notified if the course from the transfer school is equivalent to the SFSU course. If it is, you can add the SFSU course to receive credit.
Intended Users	This will be for both registered and unregistered users. You can access the course equivalency website to find an equivalent course.
URL	http://ec2-52-53-211-193.us-west-1.compute.amazonaws.com/equivalencies

Usability Metrics: Effectiveness

Task	% Completed	Errors	Comments	% Time Completed
Change an editable degree requirement	80%	Course didn't change after selecting	The pencil icons made it very easy to distinguish and alter the editable degree requirements.	
Search for a course at SFSU	0%	Pressing enter or clicking the search icon does not do anything.	The search bar location is ideal. The prompt text is very helpful.	
Use the roadmap to plan courses by semester	0%	Page under construction.	Does not support roadmap.	
Check prerequisites for an upcoming course	100%	None	Checked prerequisite easily.	
Check a course equivalency	0%	Page under construction.	Does not support equivalencies.	

Usability Metrics: Efficiency

Task	Time	Effort	Content/Design
Change an editable degree requirement	13s	Low	3 clicks
Search for a course at SFSU	12s	Low	1 click
Use the roadmap to plan courses by semester	N/A	N/A	1
Check prerequisites for an upcoming course	12s	Low	1
Check a course equivalency	N/A	N/A	1

Questionnaire: Scale 1 (Strongly Disagree) to 5 (Strongly Agree)

1. It was simple to pick a course for a degree requirement.
2. When choosing a course, the dialog was easy to navigate.
3. When choosing a course, there was plenty of information about each one.
4. It was easy to understand which requirement I was editing.
5. The course search provided me with plenty of information about any course.
6. The search feature was responsive and felt natural to use.
7. The search feature gave me enough freedom to search courses the way I wanted.
8. The roadmap was simple and easy to use.
9. The roadmap made it easy to organize my remaining semesters.
10. The roadmap was flexible enough to handle every scenario I needed it to.
11. It was easy to see what prerequisites I needed for my remaining courses.
12. The information on the prerequisite dialog was easy to understand.
13. It was clear when prerequisites were needed for any of my upcoming courses.
14. The course equivalency page was easy to use.
15. The course equivalency page had sufficient options for courses and schools.
16. The course equivalency page displayed information clearly.

See next 3 pages for copy of google form.

Google Form Link: https://forms.gle/QrA18fcUXc7JXvVG9

User Satisfaction Questionnaire

This questionnaire will test each of the functions outline in the usability test plan of csc648 milestone 04 task 02.

http://ec2-52-53-211-193.us-west-1.compute.amazonaws.com/

* Required

It was simple to pick a course for a degree requirement. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

When choosing a course, the dialog was easy to navigate. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

When choosing a course, there was plenty of information about each one. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

It was easy to understand which requirement I was editing. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The course search provided me with plenty of information about any course. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The course search provided me with plenty of information about any course. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The search feature was responsive and felt natural to use. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The search feature gave me enough freedom to search courses the way I wanted. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The roadmap was simple and easy to use. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The roadmap made it easy to organize my remaining semesters. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The roadmap was flexible enough to handle every scenario I needed it to. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

It was easy to see what prerequisites I needed for my remaining courses. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The information on the prerequisite dialog was easy to understand. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

It was clear when prerequisites were needed for any of my upcoming courses. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The course equivalency page was easy to use. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The course equivalency page had sufficient options for courses and schools. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

The course equivalency page displayed information clearly. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

Submit Clear form

QA Test Plan

In order to test the product, we will pick the following five nonfunctional requirements.

1. Every student shall get the information about the subject and their prerequisite courses and also the student shall see if there are any second level prerequisite courses.
   Test Objectives: - To test the feature we have to pass the right credentials of the user to check the all the courses and prerequisite courses are coming correctly
   HW and SW setup: - Window 11, Mac Monterey, Chrome, Firefox
   Feature to be tested: - All the courses, prerequisite courses, second level prerequisite courses.
   QA Test Plan: -

#	Description	Input	Output	Result
1	Check for all courses	Access token of login	CSC848…etc.,	PASS
2	Check for the prerequisite courses	CSC848	CSC413	FAIL
3	Check for the second level prerequisite courses	CSC848	CSC317	FAIL

2. Students can see all the basic knowledge about the course while going with the particular course.
   Test Objectives: - in order to test the objective is to check while a student enters a course it should come with right course information.
   HW and SW setup: - Window 11, Mac Monterey, Chrome, Firefox
   Feature to be tested: - Course details
   QA Test Plan: -

#	Description	Input	Output	Result
1	Undergraduate level course details	CSC 413	Units Hold, Description, Professor List	PASS
2	Graduate Level Course details	CSC 317	Units Hold, Description, Professor List	FAIL
3	Prerequisite Course	CSC 848	Units Hold, Description, Professor List	PASS

3. At a time one subject can be a prerequisite for one or more courses.
Test Objectives: - The objective of the test case is that one course can be prerequisite for one or more courses
HW and SW setup: - Window 11, Mac Monterey, Chrome, Firefox
Feature to be tested: -
QA Test Plan: -

#	Description	Input	output	Result
1	Check for one course	CSC 413	CSC 848. CSC 868	PASS
2	Check for one course	CSC 415	CSC 780, CSC 845	PASS

4. All the students who are accepted are eligible to access the tool and can view their course curriculum.
Test Objectives: - The objective to test this to verify the outsider or those students have not accepted can't access the portal.
HW and SW setup: - Window 11, Mac Monterey, Chrome, Firefox
Feature to be tested: -
QA Test Plan: -

#	Description	Input	output	Result
1	With Accepted student	Username, password	Logged in with session	FAIL
2	Without Accepted admission student	Username, Password	Not able to Access the portal	FAIL
3	Outsider with false	Username, Password	Not able to Access the portal	FAIL

5. Students can use the degree planner feature that shall give them ideas about the time expected to complete the degree.
Test Objectives: - The objective of the test case is that it is showing a perfect degree roadmap
HW and SW setup: - Window 11, Mac Monterey, Chrome, Firefox
Feature to be tested: - Degree Planner
QA Test Plan: -

#	Description	Input	Output	Result
1	To test undergraduate degree planner	All the selected courses for degree	4 years	FAIL

2	To test graduate degree planner	All the selected courses for degree	2 years	FAIL

Code Review

Part A

In our product we used web development with javascript as a primary language and used nodejs as a runtime.For frontend we used react js library which gave flexibility using html as a JSX.About the code structure is built on the atom structure where atom is component that can be used by other component which are molecules can be used as a template to make html page which is running on single div.In backend we used bob martin architecture where our request will be served with less time and maintainable code that can used further to break down microservice divide the module across the product.

Part B

Code Files that underwent review:

https://github.com/sfsu-joseo/csc648-848-06-sw-engineering-fall-2022-Team06/blob/master/Milestones/M4/BetaPrototype/client/src/components/CourseSelectionDialog.js

https://github.com/sfsu-joseo/csc648-848-06-sw-engineering-fall-2022-Team06/blob/master/Milestones/M4/BetaPrototype/client/src/components/PrerequisiteDialog.js

1. Team 7 Review
   === CourseSelectionDialog ===
   Needs comments!
   Generally you should not use inline styling excessively. Instead, use a .css file for styling because it lets you reuse styles.

   Line 21 could use a JSDoc comment describing the purpose of this function.
   e.g. /** ... */

   The ternary operators in this file should be replaced with a function or something because it is hard to read.
   For example:

```
function line78() {
        if (showDetails) {
                return (
                        <Button onClick={} ...>
                                Back
                        </Button>
                );
        } else return undefined;
}
```

   And just call this function on line 78 instead of doing all your conditional checks in the return() block.
   Same goes for line 89's very long ternary statement.

Line 94-102: What's going on here?
Line 138: This line is way too long. You should break it up into multiple lines.

=== PrerequisiteDialog ===
Needs comments.
Generally you should not use inline styling excessively. Instead, use a .css file for styling because it lets you reuse styles.

Line 15: CourseListItem could use some comments.

Need header files and in-line comments!

2) Team Member Vivek

The code in the file PrerequisiteDialog.js and CourseSelectionDialog.js is written with the ES6 modules and maintained with the latest react version which is 17.1. In the code arrow function is used to return JSX multiple times but with the dynamic values. JSX are well formed divided. The component is made on the functional base component which is the good thing to have.

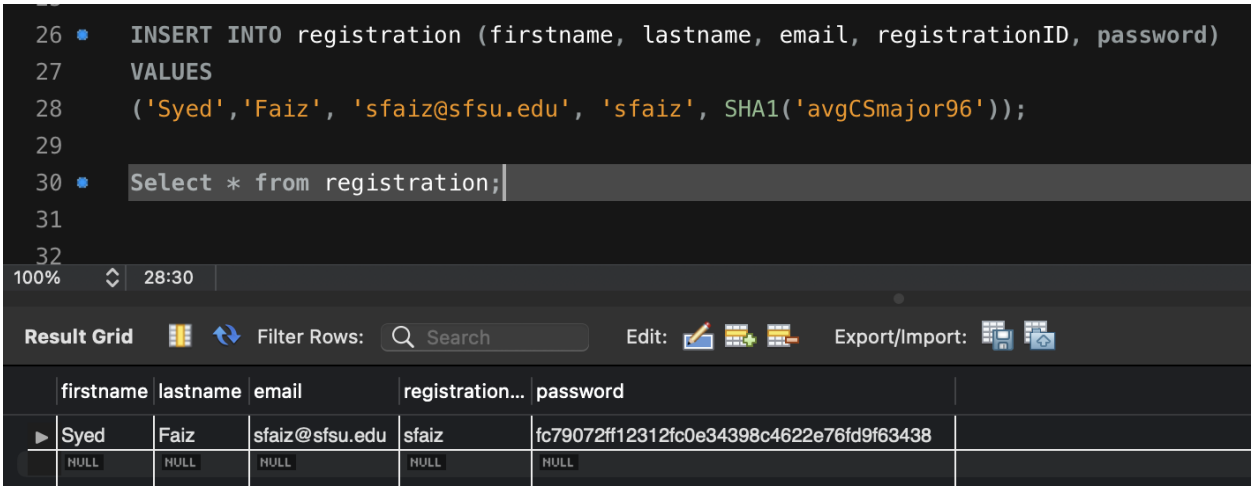
3) Code Review

The reason why I chose these 2 files is because the majority of our use cases and priority will deal with this code, and I wanted to get feedback on them. At the moment, the code has the data hardcoded, so it doesn't really help. However, the overall usability and functions will be important and I wanted to review these codes.

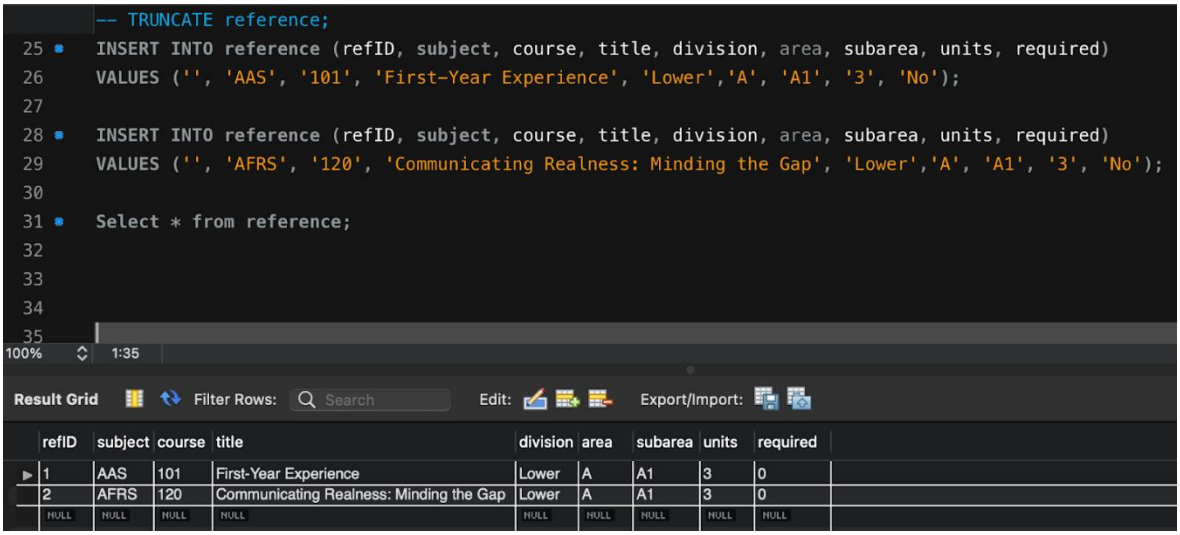
Security Practices

The main asset we are protecting for our application is the user login credential more specifically the password. To use the roadmap feature, users must enter their login which requires their registration ID and password. To protect the user password in the database, we will use the SHA1 hash function for encryption.

To show password encryption in the database, I added an insert to the registration table to register a user and use SHA1 for password. As a result, when I run the select * from registration, it returns the user account I created and the hashed encrypted password.



Similarly, for the search feature I used insert statements to add course data into the reference table. There will be over 800 courses to allow for ample search opportunities and filtering. Search could be filtered by subject, course number, title, units, division, area, subarea, and whether they are required or optional.



Non-functional Requirements

Non-Functional Specs	Status (Done, On-Track, or Issue)	Comments
All types of users will have the credentials that are provided by their university in order to access the req checker.	On-Track	We are currently undergoing this function in our current build of the website.
All the professors have to be authenticated by the university to use this tool as a university internal tool with the role of professor.	Issue	This will not be one of the main focus since it is priority 2.
All the admins have to be authorized by the university admin office in order to access this tool.	Issue	The function will not be implemented at this time for being a second priority.
All students(users) after the accepted admission are eligible to access the tool and can view their course curriculum.	On-Track	This is currently being worked on and will be completed int the future.
Every student shall get the information about the subject and their prerequisite courses and also the student shall see if there are any second level prerequisite courses.	Done	Each course shows the prerequisites need to be taken.
Every student can choose the subject and can send the request to the professor to let them in their courses.	Issue	Professor page will not be implemented due to being priority 2.
Every student can send the 2 types of requests to the professor which shall let them in the course, or they can request to eliminate prerequisite courses to eliminate them as per the student experience.	Issue	Professor page is not implemented for messages.
Students can share additional information about the work he has done with approval requests to the processor.	Issue	Professor page is second priority.
Students can also request to eliminate prerequisite courses if they have enough work done related to the course.	On-Track	This is going to be part of course equivalencies.
Students have to give mandatory notes with the reason why they want to take the particular subject.	Issue	Requires a professor page to receive message.
Students shall have ideas about all pending requests that they have sent to the professor.	Issue	Professor cannot receive request.
Students shall see the result of acceptance and rejection from the professor and can distinguish between the professor if a student has applied the same course from the two professors.	Issue	Request cannot be made due to professor page is not implemented.
Professor shall have the idea about all the requests through the total request table.	Issue	Professor table is priority 2 and will not be implemented.
Professors can directly accept or reject student requests with both types of requests.	Issue	Professor page cannot not view request.
Professor shall see the additional details along with student requests.	Issue	Professor page is second priority.
Professor can see all the numbers like waiting students, enrolled students and more.	Issue	Professor page cannot view student information.
Professor has all the rights to accept or reject the application of the student according to the details provided by the student	Issue	Professor online page still not implemented
Students can use the degree planner feature that shall give them ideas about the time expected to complete the degree.	On-track	This function is currently being implemented and have a prototype.
After matching all the credits and all the subjects' planners shall show the roadmap to take the courses and can save those to use letter on.	On-Track	We have all credits and subject's roadmap is under construction.
Students can see the list of their approved rejected courses by the professor.	Issue	Professor page is not implemented.
Users will only choose the website if they want to make the request for the approved courses and can see the degree planner.	On-Track	At the moment this is possible but there are still some tweaks that need to be made.

Non-Functional Specs	Status (Done, On-Track, or Issue)	Comments
All types of users will have the credentials that are provided by their university in order to access the req checker.	On-Track	We are currently undergoing this function in our current build of the website.
All the professors have to be authenticated by the university to use this tool as a university internal tool with the role of professor.	Issue	This will not be one of the main focus since it is priority 2.
All the admins have to be authorized by the university admin office in order to access this tool.	Issue	The function will not be implemented at this time for being a second priority.
All students(users) after the accepted admission are eligible to access the tool and can view their course curriculum.	On-Track	This is currently being worked on and will be completed int the future.
Every student shall get the information about the subject and their prerequisite courses and also the student shall see if there are any second level prerequisite courses.	Done	Each course shows the prerequisites need to be taken.
Every User will have to agree on the term and services with following conditions that will be shown in the format.	Done	This has been implemented in the login/register section.
To report any issue on the website, the user will fill the contact us page with their little summary about their concern.	On-Track	The contact page is currently being built.
Outside users can have to face legal action against the respective university.	On-Track	We are adding a legal disclaimer to the website at the bottom link.
In order to use the website student users must have their past academic info in their portal otherwise It can fail to give you the expected data.	On-Track	Most features are implemented but data is still being worked on.
Students shall be able to see a roadmap of all existing and future courses that must be taken.	On-Track	The roadmap is currently being worked on by devs.
Student shall pick their courses directly instead of manually inputting a course number	On-Track	This is currently implemented but still needs to be polished.
Students shall be able to switch between Fall, Winter and Summer courses.	On-Track	Semesters will be implemented for the course page.
Students shall be able to see the list of the subject that is approved by the professor.	Issue	A list of classes has been created but the professor page has not been created.
Student shall be authenticated after he gets admission in the university.	On-Track	Authentication is being worked on.
Students shall have ideas about all the subjects and grading methods.	On-Track	All courses will be given a description of grading options.
Students can see all the basic knowledge about the course while going with the particular course.	Done	Courses display all information inf the course section of the website.
Professor can see the list of the students who took the course and also track the record of getting dropped students.	Issue	The professor page is a second priority so it cannot be implemented.
At a time one subject can be a prerequisite for one or more courses.	Done	Prerequisite courses have been created for each course in the course section.
Students can ask more details about the ideal course and course structure to the professor before beginning with classes.	Issue	Messages cannot be sent to professor due to the professor page being second priority.
In the list students can have more idea about the courses once they decide to choose in different frames in the tool.	Done	This has been implemented in the course section of the website.
Professor shall also provide the details about the TA and other general info about their work which attracts students to take the course.	Issue	Professor page is not implemented.

Team Contributions

To be filled by team lead only:

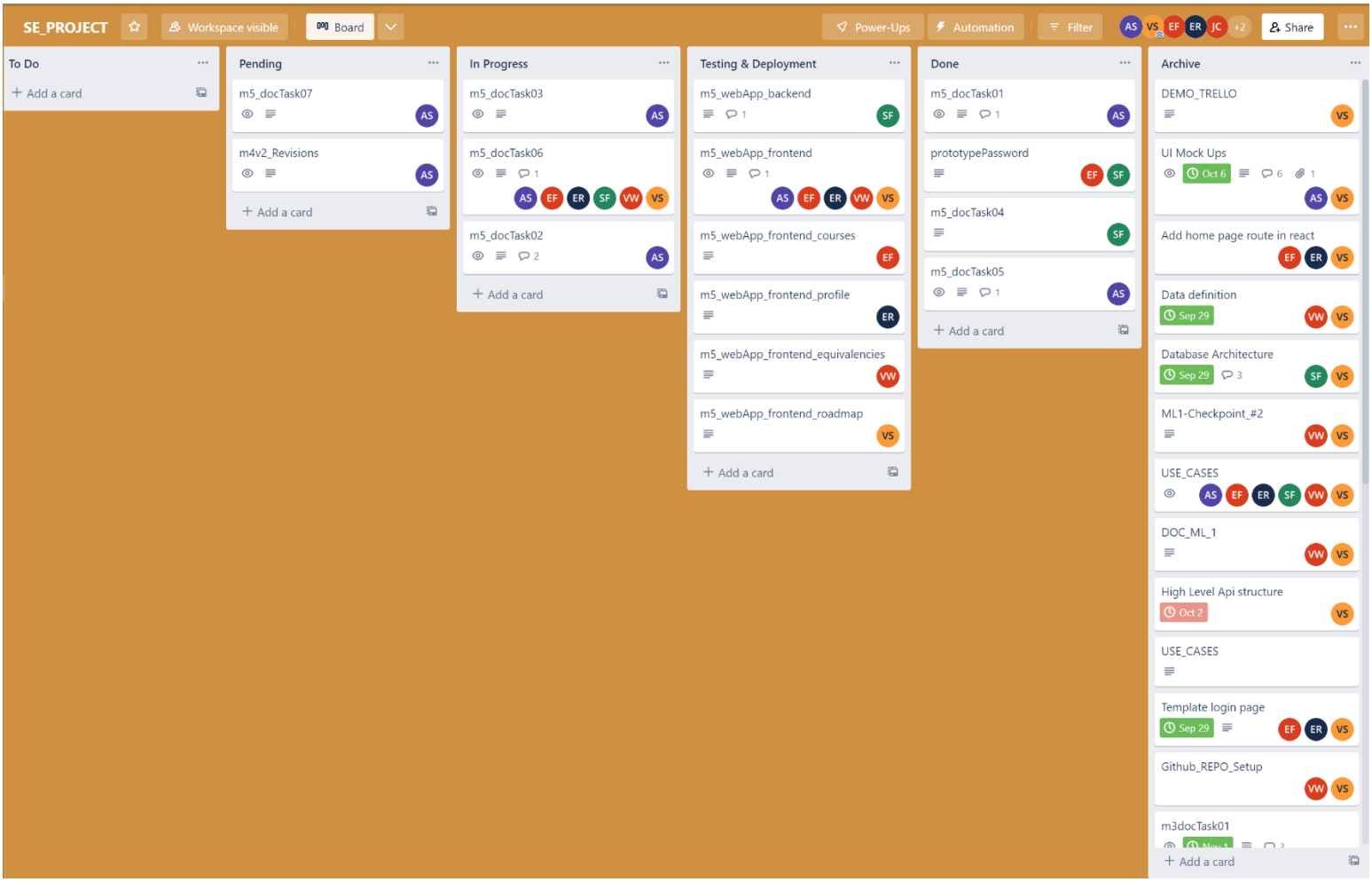
Team Member Name	Role	Rating
Syed Faiz	Backend Lead	9/10
Victoria Wilson-Anumudu	GitHub Master	9/10
Eric Falk	Frontend Lead	9/10
Vivek Santoki	Frontend	8/10
Erik Rodriguez	Frontend	9/10

Database Tables Screenshot

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
DROP SCHEMA IF EXISTS `mydb` ;

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `registration`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `registration` ;

CREATE TABLE IF NOT EXISTS `registration` (
  `firstname` VARCHAR(60) NOT NULL,
  `lastname` VARCHAR(60) NOT NULL,
  `email` VARCHAR(60) NOT NULL,
  `registrationID` VARCHAR(60) NOT NULL,
  `password` VARCHAR(75) NOT NULL,
  PRIMARY KEY (`registrationID`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `courses`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `courses` ;

CREATE TABLE IF NOT EXISTS `courses` (
  `courseID` VARCHAR(60) NOT NULL,
  `title` MEDIUMTEXT NOT NULL,
  `division` VARCHAR(60) NOT NULL,
  `area` VARCHAR(60) NOT NULL,
  `subarea` VARCHAR(60) NOT NULL,
  `unit` VARCHAR(60) NOT NULL,
  `description` LONGTEXT NOT NULL,
  PRIMARY KEY (`courseID`))
 -- UNIQUE INDEX `courseID_UNIQUE` (`courseID` ASC) VISIBLE,
 --  CONSTRAINT `registrationID`
 -- FOREIGN KEY (`courseID`)
--  REFERENCES `registration` (`registrationID`)
--  ON DELETE CASCADE -- make this cascade
-- ON UPDATE CASCADE) -- make this cascade
ENGINE = InnoDB;
-- Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails
-- (`mydb`.`courses`, CONSTRAINT `registrationID` FOREIGN KEY (`courseID`) REFERENCES `registration` (`registrationID`))

```

```
56
57  -- -----------------------------------------------------
58  -- Table `enrollment`
59  -- -----------------------------------------------------
60  DROP TABLE IF EXISTS `enrollment` ;
61
62  CREATE TABLE IF NOT EXISTS `enrollment` (
63    `enrollmentID` VARCHAR(60) NOT NULL,
64    `courseID` VARCHAR(60) NOT NULL,
65    `semester` VARCHAR(60) NOT NULL,
66    `year` YEAR NOT NULL,
67    `inprogress` TINYINT NOT NULL,
68    `completed` TINYINT NOT NULL,
69    `grade` VARCHAR(60) NULL,
70    PRIMARY KEY (`enrollmentID`, `courseID`),
71    INDEX `courseID_idx` (`courseID` ASC) VISIBLE,
72    CONSTRAINT `enrollmentID`
73      FOREIGN KEY (`enrollmentID`)
74      REFERENCES `registration` (`registrationID`)
75      ON DELETE NO ACTION
76      ON UPDATE NO ACTION,
77    CONSTRAINT `courseID`
78      FOREIGN KEY (`courseID`)
79      REFERENCES `courses` (`courseID`)
80      ON DELETE NO ACTION
81      ON UPDATE NO ACTION)
82  ENGINE = InnoDB;
83
84
85  -- -----------------------------------------------------
86  -- Table `requirement`
87  -- -----------------------------------------------------
88  DROP TABLE IF EXISTS `requirement` ;
89
90  CREATE TABLE IF NOT EXISTS `requirement` (
91    `reqID` VARCHAR(60) NOT NULL,
92    `codeID` VARCHAR(60) NOT NULL,
93    `category` VARCHAR(60) NOT NULL,
94    `exact` VARCHAR(60) NULL,
95    `group` VARCHAR(60) NULL,
96    PRIMARY KEY (`reqID`, `codeID`),
97    INDEX `courseID_idx` (`codeID` ASC) VISIBLE,
98    CONSTRAINT `reqID`
99      FOREIGN KEY (`reqID`)
100     REFERENCES `registration` (`registrationID`)
101     ON DELETE NO ACTION
102     ON UPDATE NO ACTION,
103   CONSTRAINT `codeID`
104     FOREIGN KEY (`codeID`)
105     REFERENCES `courses` (`courseID`)
106     ON DELETE NO ACTION
107     ON UPDATE NO ACTION)
108 ENGINE = InnoDB;
109
110
111 -- -----------------------------------------------------
112 -- Table `areadescriptions`
113 -- -----------------------------------------------------
114 DROP TABLE IF EXISTS `areadescriptions` ;
115
116 CREATE TABLE IF NOT EXISTS `areadescriptions` (
117   `section` VARCHAR(60) NOT NULL,
118   `info` MEDIUMTEXT NOT NULL,
119   PRIMARY KEY (`section`))
120 ENGINE = InnoDB;
121
122
123 SET SQL_MODE=@OLD_SQL_MODE;
124 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
125 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
126
```

Trello Screenshot



Team Member Contributions

Milestone 05

Team Member Name	Role	Rating
Syed Faiz	Backend Lead	10/10
Victoria Wilson-Anumudu	Github Master	6/10
Eric Falk	Frontend Lead	10/10
Vivek Santoki	Frontend	6/10
Erik Rodriguez	Frontend	6/10

Overall Rating for Milestone 1 - 5

Team Member Name	Role	Rating
Syed Faiz	Backend Lead	45/50 → 90/100
Victoria Wilson-Anumudu	Github Master	33/50 → 66/100
Eric Falk	Frontend Lead	39/50 → 78/100
Vivek Santoki	Frontend	34/50 → 68/100
Erik Rodriguez	Frontend	58/50 → 58/100

Overall Semester

Team Member Name	Role	Rating
Syed Faiz	Backend Lead	9/10
Victoria Wilson-Anumudu	Github Master	6.6/10
Eric Falk	Frontend Lead	7.8/50
Vivek Santoki	Frontend	6.8/50
Erik Rodriguez	Frontend	5.8/50

Note: to keep a consistent grading rubric, I kept the grading to 50% documentation and 50% prototype coding throughout the milestone grading for team members who was given task in both. The given final semester score reflects this performance on both documentation and prototype equally.

1. **Syed Faiz 9/10**
   - 1.1. Documentation
     - 1.1.1. Contributes fully to the documentation and does all the require work outline in the assigned documentation task. I would not need to inform or remind him to do anything. When given task, will work on it and provide updates on progress.
   - 1.2. WebApp
     - 1.2.1. Primarily responsible for the backend data and hosting it on AWS. Any issues or problems were taken care of and was notify. When given task, done on a timely manner, and will work on it and provide updates on progress.
   - 1.3. Comments
     - 1.3.1. Syed can do the work and has contributed fully throughout the semester. If there was any questions or problems regarding the backend, I can rely on him to either find a solution, or seek answers on his own, whether it be research or asking CTO Ortiz. Participates in using Trello and providing progress and updates. Takes initiative and provides recommendations and not afraid to give criticize or opinion. Full participation and communication. Had missed some lectures but notified of absence.
   - 1.4. Feedback
     - 1.4.1. There was at times when I needed to get information or have a question on the backend, and I would have to either schedule a time or wait for a reply where we can discuss said this; there was usually a delay of some hours to get this information. This would sometimes result on the frontend needing to wait until I have the information/answer. However, even with a delay, Syed would pull through and provide the necessary information/solution, with or without a delay.

2. **Victoria Wilson-Anumudu 6.6/10**
   - 2.1. Documentation
     - 2.1.1. Contributes fully to the documentation and does all work outline in the assigned and given tasks. Although will need reminders and guidance on task, provided comments and feedback on documentation when encountering problems.
   - 2.2. WebApp
     - 2.2.1. In terms of coding, there was not any contributions nor much involvement in the webapp. Only coding done was on the equivalency webpage with help from frontend lead Eric.
   - 2.3. Comments
     - 2.3.1. Victoria can do the work and has contributed throughout the semester. Though there wasn't much initiative in participating in more then what was assigned, she does carry out the task when assign or needs corrections. Communication on progress and working on a timely manner needs to be improved as well as her coding skills. Had missed some lectures but notified of absence.
   - 2.4. Feedback
     - 2.4.1. Would need to take more initiative in participating overall. Although communication is great when engaged, it becomes sparse on either Discord, Zoom, or in person; it's also difficult to ascertain task progress without requesting it. In terms of coding, will need to improve upon as well as delivering given task on time; needs to improve on providing enough time to do assign work.

3. Eric Falk 7.8/10

3.1. Documentation

3.1.1. Contributes fully to the documentation and does all work outline in the assigned and given tasks. I would not need to inform or remind him to do work. Although I needed to request updates on progress.

3.2. WebApp

3.2.1. Eric has coded the majority of the webapp and has done great work on building and maintaining the webapp. We have been using derivatives of his original webapp/prototype for our project, adding slowly to it throughout this semester. From our documentation, his coding skills was able to implement what was outline and requested. I have deferred all frontend judgment and ideas to him; has taken initiative in carrying out extra work and ideas in refining and improving without being told too.

3.3. Comments

3.3.1. Eric can do the work and has contributed greatly throughout the semester. Though in the beginning he wasn't fully participative, he took initiative in contributing more then what was assign about a month into the semester and being fully active and communicative in person or over Discord or Zoom ever since. Provides updates on work progress daily; can rely on him to work independently with great results. Takes initiative and provides recommendations and not afraid to give criticize or opinion. Has not missed a class lecture.

3.4. Feedback

3.4.1. Eric has shown great work and has been fully participative and communicative. Although work has been done in a timely manner, will need to improve on his scheduling in terms of schedule meetups.

4. Vivek Santoki 6.8/10

4.1. Documentation

4.1.1. Contributes to the documentation, however he is not consistent with a schedule in implementing task. Needed to be given reminders to carry out assigned task and will always need to go over his work to confirm completion of task objectives.

4.2. WebApp

4.2.1. In terms of coding, it is difficult to ascertain if Vivek has coding skills applicable to the full stack technology. When given coding task, work was late; when requested to collaborate with other team members, hardly any involvement nor communications. Given his experience in the tech industry, I am surprise Vivek has not shown any initiative nor evidence of either providing, supporting, or consulting this team in terms of coding with either frontend or backend software technology. Only when requested in person, would he give good opinions and recommendations only; when requested to meetup in person to go over UX/UI did he give great information and logic, though no coding was involved.

4.3. Comments

4.3.1. Vivek can do the work but would need to be constantly reminded to do so. Had to consult with CTO Ortiz at one point regarding lateness of work. When given feedback on improving, Vivek would not carryout improvements; this resulted in uncertainty in his role and responsibilities. Vivek has missed class lecture a few times with notifications.

4.4. Feedback

4.4.1. Uncertain in what feedback to give since I had mentioned several things he could improve on as a team member throughout this semester, and they were not improved upon. The uncertainty in completing assigned task, or meeting deadlines stands out as the primary objective he will need to improve on if he is to continue working in this line of field. Furthermore, he will need to take more initiative in participating and communicating with not just the team lead, but with other team members. Vivek will need to improve both hard and soft skills to succeed after this project. Rather than give constant reassurance of improvements to team lead, Vivek needs to give himself constant reminders in himself in carrying out the work that needs to the done based on this project experience.

5. Erik Rodriguez 5.8/10

5.1. Documentation

5.1.1. Contributes fully to the documentation and does all the work outline in the assigned and given task. Sometimes would need to request updates on progress.

5.2. WebApp

5.2.1. In terms of coding, there was not any contributions nor much involvement in the webapp. Only coding done was on the profile webpage.

5.3. Comments

5.3.1. Erik can do the work and has contributed throughout the semester. Though there wasn't much initiative in participating in more then what was assigned, he does carry out the task when assign or needs corrections. Communication on progress and working on a timely manner needs to be improved as well as his coding skills

5.4. Feedback

5.4.1. Would need to take more initiative in participating overall and being communicative. Although communication is great when engaged, it becomes sparse on either Discord, or Zoom. In terms of coding, will need to improve upon as well as delivering given task on time; needs to improve on providing enough time to do assign work.

Post Analysis

Main Challenges

On of the main challenges was communication between team members. In the beginning, there was hardly any communication, and each of us individually were not communicating enough. Some team members were communicating more then others, and in a way is still a challenge. Even though we had means of communication through Discord, Zoom or in person, I always found it a challenge to have a team member being active in communicating with me on their work progress. It was only when I engaged my team members is when I can receive their updates.

Another challenge I experience is meeting deadlines and getting progress done on time. When team members were given task to do, it would be difficult to ascertain the length or the estimated time it'll take for the task to be finished. Usually it'll be near the deadline, and the submitted work would sometimes need to be reviewed and corrected. I admit I am guilty of theses, and let personal problems get in the way of getting work done. I had to prioritize this academic work and be able to guide my team to getting work done. Nevertheless, being in this course required acceptance of completing the course work and doing your best.

Another challenge I faced was having our group in-sync in terms of collaboration. Throughout the semester as team lead, I always felt there was no synchronization in working together and trying to solve or understand problems or issues. In the beginning we did have some synchronization, being unified on google docs and working on it over Discord. However, as the semester progressed, our communication relapsed, and in a way, we drifted away from our group; only during lecture when we would briefly talk to each other then go our own way. Then proceed to carry out our assigned task, and then add it google doc.

Finally, the last challenge was having a concrete vision for our webapp. I realized throughout the semester we didn't really have a concrete vision of how our webapp will be or how it should work. We had an idea, and it was vague; it wasn't until it was molded through various milestones documentation that we were able to see what how it works. Through many comments, drawings, feedback, and redoes were we able to get the current webapp project. Each of us in the team had different ideas, and it was a challenge trying to either incorporate or add them in some way to our webapp. We had many priorities that were demoted due to how much we wanted this webapp to be revolutionary.

What would I do better next time to address these main challenges

The way I approached in addressing the communication issue was to request team members to collaborate with other team members. By doing this, I wanted to break the barrier of our team being comfortable enough to rely on others for help if need to. Although some team members were working full time, Discord helped us provide an adequate place to communicate with each other if there were to be a delay. However, I could do only so much, and I wanted each team members to take the initiative to be more communicative with each other rather then needing to be engaged in order to be communicative.

The challenge to meeting deadlines and getting progress done has been address; we used Trello in order to map our progress and see our updates. However, what I would have done better was to require team members to be active in updating Trello and providing everyone a visual progress of the overall teamwork. It was usually me requesting updates on team members task and updating Trello manually. If I had required participation as well as

clarifying what the results should be, our progression and completion of our over group project would have been easier to map out.

What I would have done better in syncing our collaboration is carrying out the advice given above; I would have needed to address the communication and collaboration issue. I would have provided a better means of trying to get team members to communicate and require them to do so. Rather then assigning everyone a task, I would've have required more collaboration in order to have everyone work with each other more often.

Finally, what I would have done better to address our concrete vision for our webapp is having our team sit done and refined what we wanted to accomplish vs what we can realistically accomplish with the individual skills we have. Each of us individually contributed to having a component to our webapp project. As a result, we had many priorities that couldn't have been implemented to due technical or time constraints. We needed to be realistic with our goals and time. We were too ambitious, and not realistic enough to notice we had a barrier we could not overcome; each of us had different levels of coding skills that resulted in very few team members contributing to coding. However, we adapted and tried to implement the objectives we can realistically accomplished.