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
M1V1	September 15, 2022

TABLE OF CONTENTS

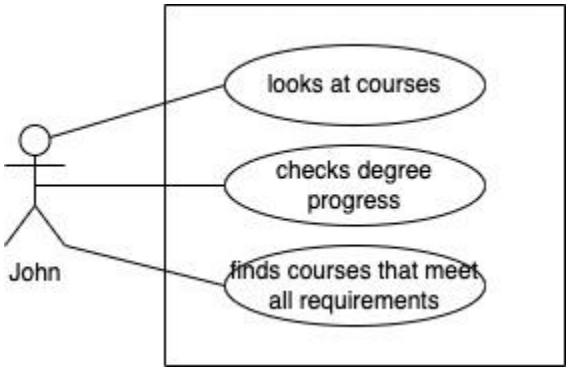
Executive Summary

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.

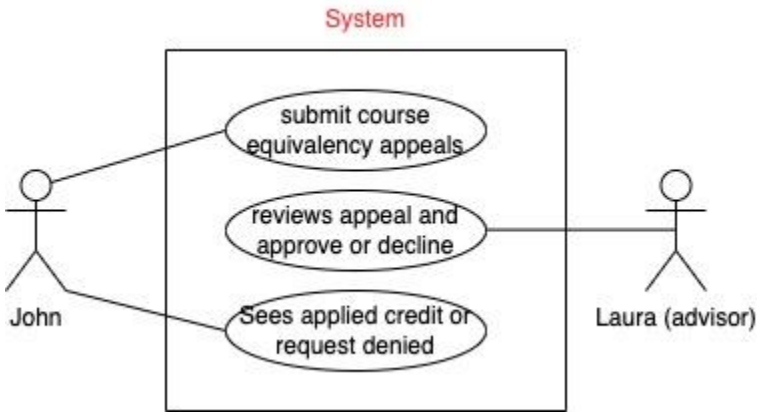
The Req Check software takes one school's course code and matches it with the other schools' courses. An automated system like Req Check will efficiently map out when a student will graduate as well as ensure they are on the right track as a new transfer student. Req Check will alleviate stress for students 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

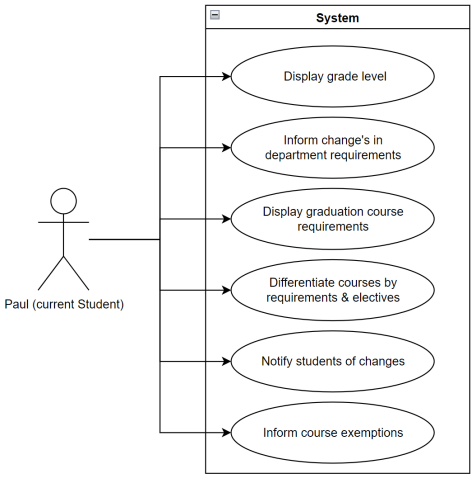
Use Case: Efficient Degree Planning
Actors: John (Student)
Description: John is a computer science student at SFSU and is excited to graduate. He is missing several requirements including Area B2, American Ethnic and Racial Minorities, Social Justice, U.S. History, and U.S. Govt. Wanting to graduate 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.



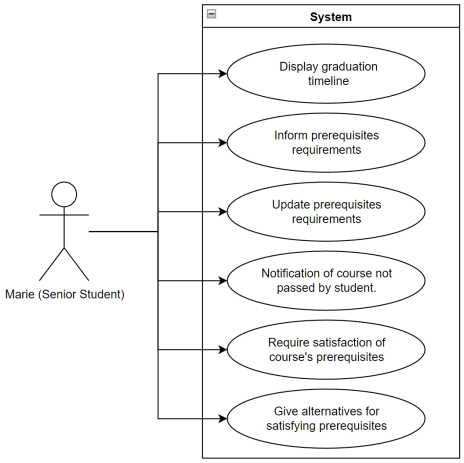
Use Case: Course Equivalency Appeal
Actors: John (Transfer Student), Chris (Transfer Student), Laura (Advisor)
Description: John, a transfer student from UNLV, logs into Req Check and sees that many of his courses do not have existing equivalencies at SFSU. He has taken calculus I and calculus II at 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.

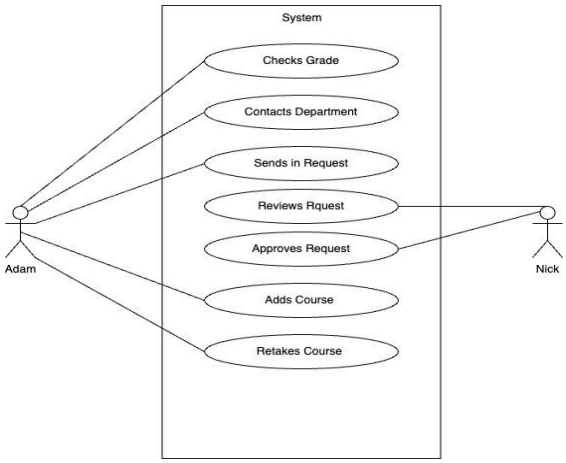


Use Case: Students satisfying prerequisites for electives
Actors: Senior Student (Marie), Prerequisite
Description: Marie has one last semester left to graduate from SFSU as a computer science major. She is currently taking a required class in the spring semester in order to satisfy the prerequisite 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.



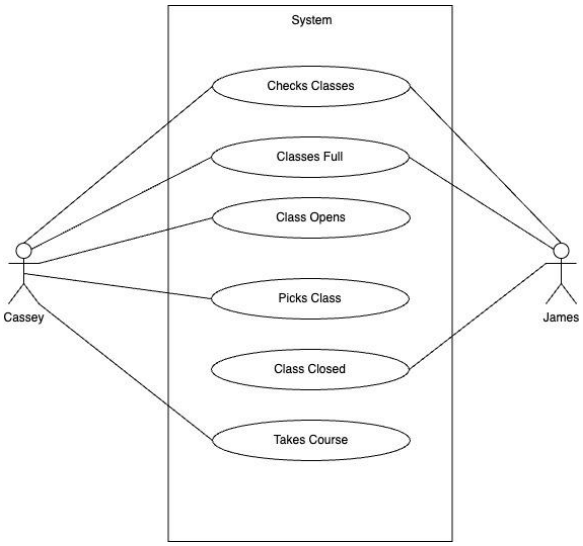
Use Case: Grade Check
Actor: Adam(Current Student) and Nick(Professor)
Description: Adam had recently taken MATH324 and received a C- but he needs at least a C to continue on to his next course, which would be CSC 415. Unfortunately, the current system will not let him retake the course since the school 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.

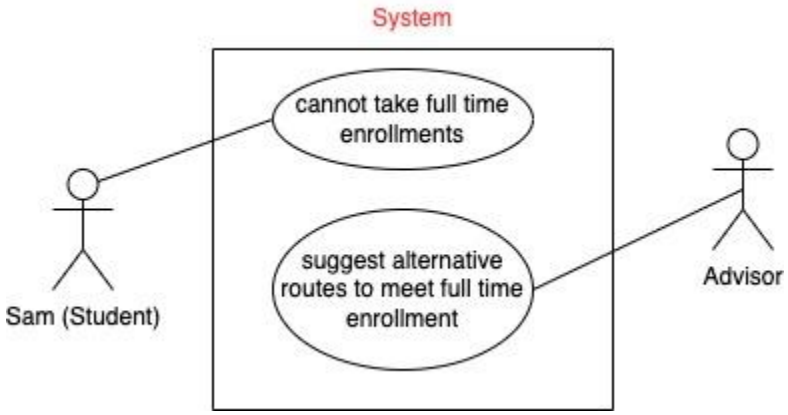
Use Case Diagram:



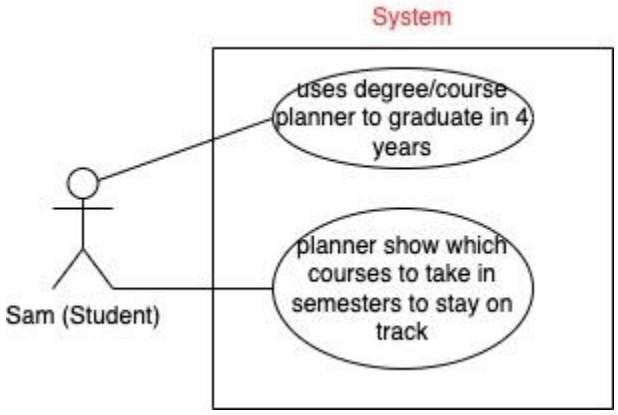
Use Case: Semester courses are offered

Actors: Sam (Transfer Student), Advisor

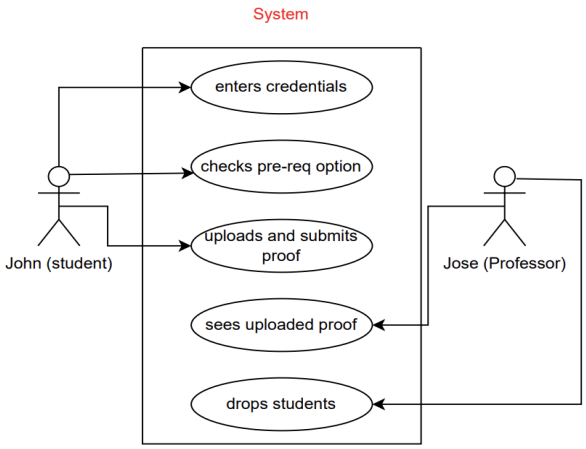
Description: Sam is a new transfer student and is trying to plan her fall semester. She wants to enroll as a full time student this year which is a minimum of 12 units. She has completed all of her general education requirements and is having a hard time making a full time schedule due to the fact that 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:

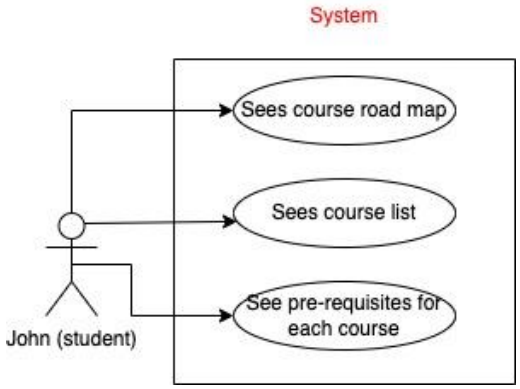


Use Case: Prerequisite Road Map Chart

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.

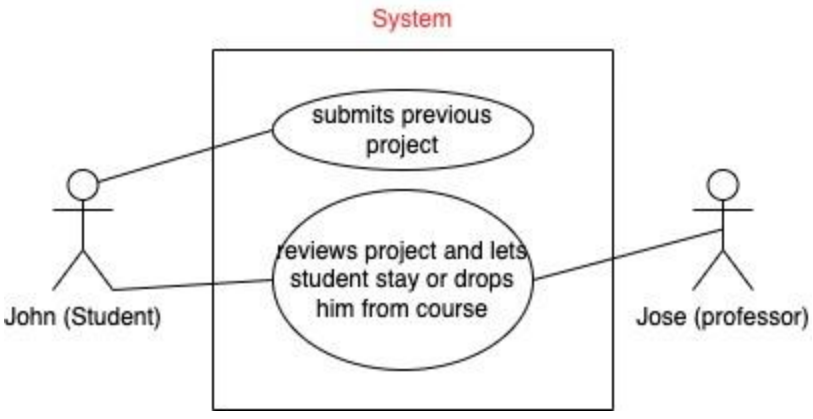
Use Case Diagram:



Use Case: Accepting or Rejecting to the course.

Actors: Jose (Professor), John (Student).

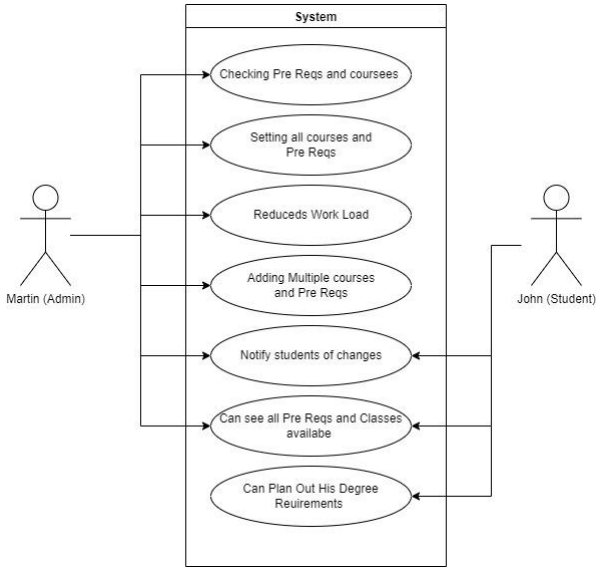
Description: Jose is the professor and John is a student. Jose wants to accept the student but wants to know a little bit more about his work for a particular project.John will give the idea about the course work while he requests the professor to include it in his course.The way Jose will have more 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).

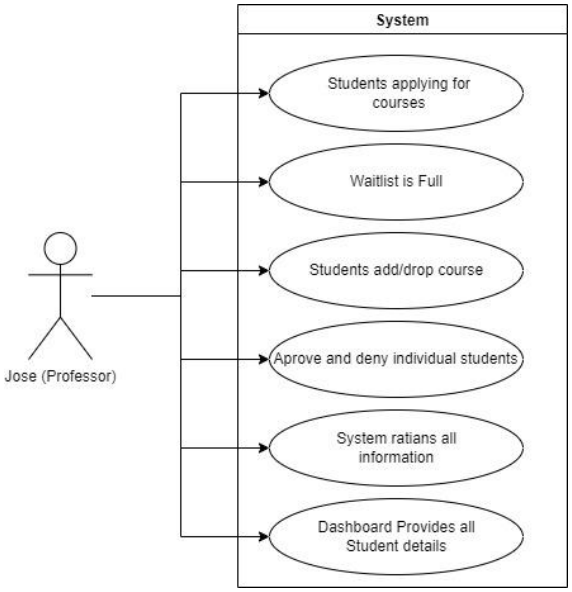
Description: As an administrator, Martin handles all the courses and prerequisite courses of the program. Using req check, Martin easily sets all the courses and their prerequisite courses reducing Martin's administrative work.He can add multiple prerequisite courses and can also add non prerequisite 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).

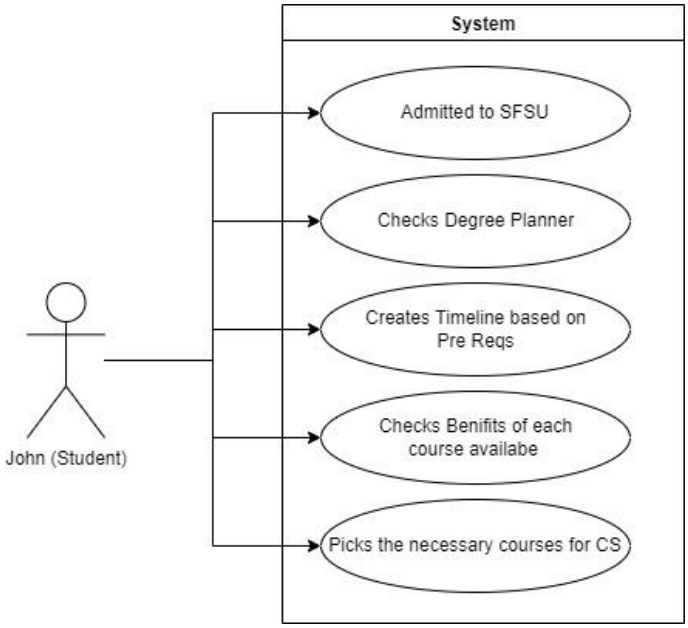
Description: Many students apply for the same course and it gets difficult to handle all the numbers for Jose for the waiting list, the number of students dropped the class,How many student he approved and their details, to short all the things req checker will help him to provide all the details on his dashboard as a professor.



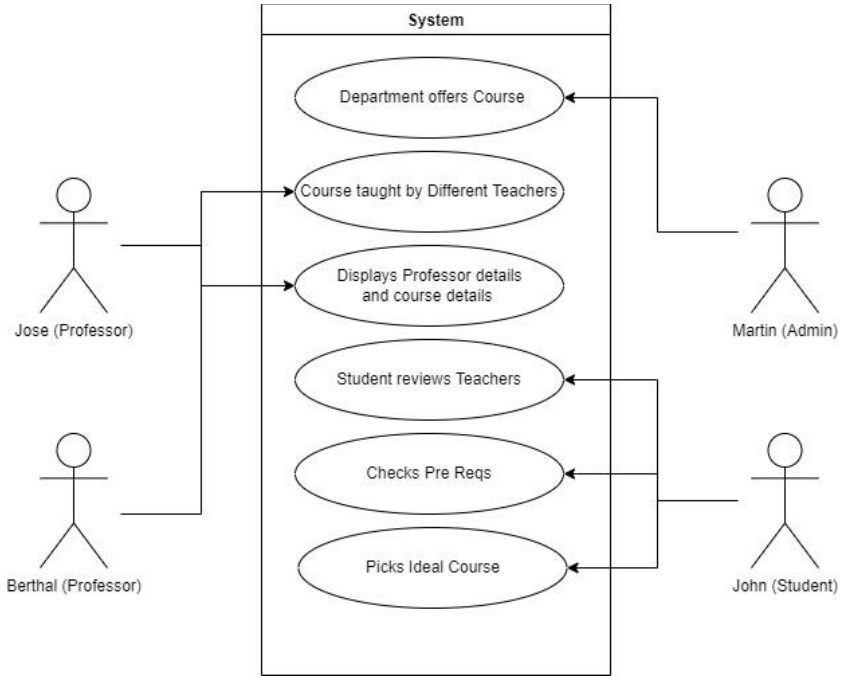
Use Case: Set Timeline for the Degree.

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.


System
Department offers Course
Course taught by Different Teachers
Displays Professor details and course details
Student reviews Teachers
Checks Pre Reqs
Picks Ideal Course
Jose (Professor)
Berthal (Professor)
Martin (Admin)
John (Student)

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

Appointment:
Student first name
Student Last name
Advisor first name
Advisor last name
Date/Time
Notes

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

Schedule
Advisor first name
Advisor last name
Time slots
Dates

Initial list of functional requirements

1. Students have to be authenticated by the university to use this tool as a university internal tool with the role of student.
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. In order to use the degree planner feature students have to fill the appropriate subjects that they want to take in the degree.

20. 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.
21. Suggestion of the timing shall depend on how much credit a particular student has to complete their degree.
22. 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.
23. Students can see the list of their approved rejected courses by the professor.
24. The application should protect the encapsulated user data and disclose information based on roles.
25. The user (if student) must be given information on whether or not a specific course is transferable to a certain school.
26. The application should automatically verify a student's academic records hence the administrator.
27. The application should be able to authenticate user identity by the university.
28. System shall provide all information on passed courses, including professor, grade, and session.
29. Students shall be able to see a roadmap of all existing and future courses that must be taken.
30. System shall send notifications of recommended courses and waitlisted courses for users to add.
31. Student shall pick their courses directly instead of manually inputting a course number
32. Students shall be able to switch between Fall, Winter and Summer courses.
33. Students shall be able to see the list of the subject that is approved by the professor.
34. Student shall be authenticated after he gets admission in the university.
35. Students shall have ideas about all the subjects and grading methods.
36. Students can see all the basic knowledge about the course while going with the particular course.
37. Professor can see the list of the students who took the course and also track the record of getting dropped students.
38. Professor only can approve a defined number of students to their courses.
39. Once any student dropped the seats shall be filled up by the professor on the waiting list queue of the students.
40. Once any student drops, the professor shall have the idea about filling up that seat.
41. Each student shall have a chance to approve a request but seats shall be filled up by the first come first serve.
42. Admin can set all the new records and can make a hierarchy of prerequisite courses.
43. Admin can also arrange all the courses department wise.
44. At a time one subject can be a prerequisite for one or more courses.
45. Admin can change any time courses and prerequisites as per the demand of the admin panel.
46. Admin can also add the prerequisite on a defined hierarchy of courses.

47. Students shall have category wise subjects that they have to choose.
48. Students can ask more details about the ideal course and course structure to the professor before beginning with classes.
49. Course shall have less dependency to any other entity.
50. In the list students can have more idea about the courses once they decide to choose in different frames in the tool.
51. Admin can have all the list of all courses and how much students have taken the courses.
52. Professor shall also provide the details about the TA and other general info about their work which attracts students to take the course.
53. Also students can see the timing of classes and if they are classing it shall not be chosen automatically by the system.
54. Students can see the list of all the professors for the same course to get better knowledge about the professor course structure.
55. The system shall display the student's grade level (current, transfer, new).
56. The system shall inform students of any change in department requirements.
57. The system shall display the course requirements for graduation.
58. The system shall differentiate courses by requirements and electives.
59. The system shall notify students of all newly required courses.
60. The system shall update all courses based on students input data.
61. The system shall inform students of course exemptions.
62. The system shall inform the student the number of semesters left to graduate.
63. The system shall inform the student any prerequisites required prior to enrolling in a course.
64. The system shall update any prerequisite requirements.
65. The system shall be notified if a course has not been passed.
66. The system shall enforce any prerequisites prior to enrolling in courses.
67. The system shall require satisfaction of prerequisites in order to enroll in courses.
68. The system shall give alternatives to satisfying prerequisites if applicable.

List of non-functional requirements

1. The Req checker will be available on the WWW platform and a person who is authenticated and respected can use the software.
2. The Website can serve the number of people at a time and the number will be defined by the college administration.
3. All types of user will have the credentials that are provided by their university in order to access the req checker.
4. As we used an optimized library to build the website so it will give the performance even if the user has low internet.
5. Users will only choose the website if they want to make the request for the approved courses and can see the degree planner.
6. Every User will have to agree on the term and services with following conditions that will be shown in the format.
7. To report any issue in the website, the user will fill the contact us page with their little summary about their concern.
8. Outside users can have to face legal action against the respective university.
9. 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.
10. 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.
11. System shall give a grade level status based on overall course completion.
12. Each requirement change shall update the overall course completion right away.
13. Each course shall be given a category of major, minor, GE, or elective.
14. Each course shall be appointed based on a student’s grade level (current, transfer, new).
15. Each exemption shall be based on course requirements and student grade level.
16. System shall give numerical status on the number of semesters left before graduating.
17. Each course shall have numerical options of prequisites.
18. Each course shall have an enrollment status based whether a grade has been given or not.
19. System shall not allow students to enroll in courses they do not qualify for.
20. System shall require students to complete prerequisites in order to enroll.
21. Each course shall display semester’s it is being offered for up to 4 years.
22. Each semester shall display the courses that can be taken prior to enrollment.
23. Main page will display the user's schedule and classes with professors name, date and class number.
24. 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.
25. System will be able to support a large number of users at one time.
26. System will run on Windows, Mac and Lynx.
27. System will be available during all hours of the day including weekends and holidays.

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
Text Search	+	+	+	++	+	+
Boolean Search	++	-	+	-	-	-
Browse	++	+	+	+	++	++
Shopping Cart	+	-	+	-	+	+
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 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	4
Erik Rodriguez	Front end	4