
Software Requirements Specification

for

Majorizer

Version 1.0 approved

**Prepared by Robert Licata, Niall Pepper, Peter Dorovitsine, Sangwon
Youn, and Tyler Yankee**

Clarkson University, Department of Computer Science

May 5, 2023

Table of Contents

Revision History	3
1. Introduction	4
Purpose	4
Document Conventions	4
Intended Audience and Reading Suggestions	4
Project Scope	4
References	4
2. Overall Description	5
Product Perspective	5
Product Features	5
User Classes and Characteristics	5
Hardware Requirements	6
Design and Implementation Constraints	6
User Documentation	6
Assumptions and Dependencies	6
3. System Features	7
3.1 User Authentication and Account Information	7
3.2 Course Schedule Builder	7
3.3 Course Catalog	8
3.4 Course History	9
3.5 Advisor/Advisee View	9
3.6 Advisor/Advisee Manager	10
4. External Interface Requirements	11
User Interfaces	11
Hardware Interfaces	12
Software Interfaces	12
Communications Interfaces	13
5. Other Nonfunctional Requirements	14
Performance Requirements	14
Safety Requirements	14
Security Requirements	14
Software Quality Attributes	14
Business Rules	15
6. Key Milestones	16
7. Key Resource Requirements	17
8. Other Requirements	18
9. Requirement Change Management	19
10. Restrictions, Limitations, and Constraints	20

Revision History

Name	Date	Reason For Changes	Version
Initial Version	5/5/2023	Submission at end of semester Spring 2023	1.0

1. Introduction

Purpose

Majorizer is a course scheduling application intended to assist students in determining their path of university study, to assist faculty in providing guidance to students, and allow administrators to coordinate these student-faculty relationships effectively.

Document Conventions

In this document, every requirement has its own priority explicitly stated.

Intended Audience and Reading Suggestions

This document is primarily intended for project managers to be able to understand Majorizer as a software product in its current state. It contains descriptions of the product, intended use cases (including diagrams for illustration), functional requirements (including features), and nonfunctional requirements (including development and operating environment, performance, and technologies used in development). It is recommended to read this document at least through Section 5 in the given order to understand the features and restraints before operating the software.

Project Scope

Majorizer is a university scheduling software. It attempts to solve an existing issue with many of these products in that planning for student course paths is often a tedious, time-consuming process that must be manually completed by students and faculty advisors. While we don't discredit the need for this system of personal advising on a case-by-case basis, we emphasize that students and advisors should be able to explore alternative options to degree completion and career planning in an easily accessible manner. Majorizer allows for this by not only providing raw information to students and advisors and allowing for communication between them — which is the general state of most current products — but also by synthesizing this information to provide a platform for students and advisors to make decisions without being hindered by small details. In that sense, Majorizer makes degree completion a more efficient and equitable process.

References

This document does not specifically refer to it anywhere, but it is worth noting that the Majorizer team website — containing progress reports, team member information, and more — can be found at: <https://webspace.clarkson.edu/~licatar/>.

2. Overall Description

Product Perspective

Majorizer is a new, self-contained product which originates from other university administrative software systems. It attempts to build upon class scheduling programs that currently exist and make a more helpful version with more features.

Product Features

Majorizer will have the following features, allowing students to efficiently plan their course schedules to ensure they meet their desired goals.

- User Authentication and Account Information: A way for users to store their information in an account with username and password for ease of use in the future
- Course Schedule Builder: A tool for both students and faculty advisors to determine the number of possible directions in assembling a course schedule.
- Course Catalog: A tool for both students and faculty advisors for accessing all courses currently offered by the university and their associated information.
- Course History: A tool for students for accessing the courses they have taken throughout their academic career and associated completion information.
- Advisor/Advisee View: A tool for students or advisors to access the current list of all of their advisors/advisees and associated information, including the nature of the advising relationship (e.g., academic, honors, research, etc.).
- Advisor/Advisee Manager: A tool for administrators to assign faculty advisors to student advisees, ensuring that all students have access to a resource for guidance.

User Classes and Characteristics

As previously mentioned, there are three anticipated user classes for Majorizer, listed in order of importance to satisfy:

- Students will utilize Majorizer as a planning tool for their degree completion. Since this is a primarily younger demographic, we expect them to more easily understand the intuitive user interface. However, since they are less experienced than the university faculty on what courses are offered, they will need to rely on the information presented through Majorizer more. Students will likely use the schedule builder feature more often than the other user classes.
- Faculty Advisors will utilize Majorizer as a planning tool to assist their students in degree completion. Advisors have a wide variety of technological experience and familiarity with university regulations, so the user interface is designed as intuitively as possible to account for this. They likely won't use the schedule builder tool very often, instead pointing students to the information presented through the other features and complementing that with their own expertise to effectively advise their students. They will likely use the advisee feature more than students will use the advisor feature, since each advisor has many more students to manage than vice versa.

- Administrators will utilize Majorizer to coordinate students with advisors, ensuring that every student has (at least) an academic advisor to guide them to degree completion and assist them with any other questions or issues they may have. Administrators have elevated privileges in that they can assign students to advisors. However, this is likely the only functionality they will use on a regular basis.

Hardware Requirements

Majorizer is being developed on Windows 10/11, but due to the frameworks it uses — Flutter, Django, and MySQL — it has no strictly-defined hardware requirements and is designed to operate across a variety of platforms.

Design and Implementation Constraints

There were no limitations on hardware or software used for this project. The only major limitation was the available time to complete the project.

User Documentation

For the current revision, no additional documentation will be provided with Majorizer. In the future, tools such as a course catalog in PDF format or a series of video tutorials showcasing how to use the software could be provided to students and faculty.

Assumptions and Dependencies

Majorizer is currently designed to only work with a limited set of majors and minors at Clarkson University. The program in its full form would allow for any set of majors and minors and ideally be usable by any university if they could upload their course catalogs. However, the current version is a proof of concept that allows for the selection of the following majors/minors:

- Computer Science
- Chemical Engineering
- Mathematics
- Psychology

3. System Features

3.1 User Authentication and Account Information

3.1.1 Description and Priority

When a user opens the Majorizer application, they will be prompted to log in, or sign up if they don't yet have an account. Users will sign in using their Google account, which will only allow users with an @clarkson.edu email account to access the software. The user's information will be stored and be used to ensure accurate information is referenced when they are using the application. This includes what type of user they are, and any advisor/advisee pairings.

This feature is essential to ensure accurate information is used to advise each student, and to ensure advisors and administrators can use the application properly. Therefore, it is a high priority feature.

3.1.2 Stimulus/Response Sequences

Upon opening the application, users will see the landing page. This landing page has a 'Log In' button as well as a 'Sign Up' button. Selecting either one of these will prompt the user to log into their Google account. When signing up, the user will be prompted to enter their class of user, and department, if applicable. If they enter a valid @clarkson.edu email address, they will continue to the home screen of the application.

3.1.3 Functional Requirements

REQ-1.1: Store user data so that return users can login and access their information

REQ-1.2: Only allow accounts to be created with an @clarkson.edu email address

REQ-1.3: Only allow users who have an account to access the application

3.2 Course Schedule Builder

3.2.1 Description and Priority

This feature will allow students to enter factors that will influence their academic career. This includes major/minor combinations, the possibility of co-ops, studying abroad, and graduating early. When inputted along with their course history, this feature will calculate a number of possible schedules for the rest of the student's academic career. This will not have any functionality in enrolling the student in any classes or changing their major - it is purely informational.

This feature encapsulates the main goal of this software, and as such it is a high priority feature.

3.2.2 Stimulus/Response Sequences

After signing in, the user can select the navigation tab in the top right of the screen. This navigation tab is accessible on every screen of the application. This will open up a list of different features to access. Selecting the 'Schedule Builder' tab will take the user to the schedule builder screen. On this screen, there are 7 dropdown menus where the user can enter up to two majors and two minors, as well as a co-op semester, a study abroad semester, or a graduation date.

Once the desired settings are selected, the 'Build Schedule' button can be pressed to yield a possible schedule. This will show a possible list of classes to take for the upcoming semester. A set of arrow buttons can navigate between the upcoming semester and future semesters.

3.2.3 Functional Requirements

REQ-2.1: Create a set of schedules that fulfill all major and minor requirements

REQ-2.2: Use provided course history within the schedule building algorithm

REQ-2.3: Allow for users to change majors/minors

REQ-2.4: Account for edge cases of study abroad, co-op, etc.

3.3 Course Catalog

3.3.1 Description and Priority

This feature will allow students to view all courses being offered at their university. This will allow them to directly see which courses they would be interested in taking and help shape their choice of future academic plan.

This feature is informational and does not assist in the main functionality of the program. However, it would be very helpful as referential material for users. Therefore, it is a medium priority.

3.3.2 Stimulus/Response Sequences

Selecting the 'Course Catalog' tab within the navigation menu will take the user to the course catalog screen. On the course catalog screen, there is a grid of large buttons that list departments along with their abbreviations. Selecting one of these buttons will take the user to a screen that lists all the courses offered by that department, along with its course code and some other information. Selecting a course will open a small tab that gives a brief description of the course along with the instructor teaching it.

3.3.3 Functional Requirements

REQ-3.1: List courses offered by the university as well as all associated information (instructor, credit hours, knowledge areas, etc.)

3.4 Course History

3.4.1 Description and Priority

This feature will allow students to input and view their course history. It will display all courses taken so far, along with the information of which term it was taken, the credits taken, the grade received, and if it was transferred or taken at the university.

The use of course history is nearly essential in the operation of this software - otherwise, the software will be very difficult to use for users that have taken multiple semesters of courses. Therefore, it is a high priority.

3.4.2 Stimulus/Response Sequences

Selecting the 'Add Transfer Credit' tab within the navigation menu will take the user to the add transfer credit screen. On this screen, the user will be able to select via dropdown menu specific courses they have taken in the past, along with all relevant information. This information will then be stored in their account.

Selecting the 'Course History' tab within the navigation menu will take the user to the course history screen. Within the course history, a list of all courses taken by the user will be shown. This will show the information stated above.

3.4.3 Functional Requirements

REQ-4.1: Read and store user course information alongside user account

REQ-4.2: Display student's course history along with all relevant information (grade, semester taken, etc.)

3.5 Advisor/Advisee View

3.5.1 Description and Priority

This feature will offer a way for users to see who their advisor is if they are a student, or see who their advisees are if they are an advisor. It will show which advising capacity each of these relationships is (academic, research, etc.). It will allow users to reach out to their advisor/advisee by providing an email address, which will help students who want assistance with their course planning. All students in the program should have at least one advisor.

This feature is nonessential to the main functionality of this program and as such is a low priority.

3.5.2 Stimulus/Response Sequences

Selecting the 'Advisor Manager' tab within the navigation menu will take the user to the advisor view screen. This screen will look different for different types of users.

For students, the advisor view screen will display a list of all the advisors a student has. It will display the advisors' name, department, advising capacity, and email address.

For advisors, the advisor view screen will display a list of all the students an advisor has. It will display the students' name, majors and minors, advising capacity, and email address.

3.5.3 Functional Requirements

REQ-5.1: Display list of advisors/advisees with all relevant information

3.6 Advisor/Advisee Manager

3.6.1 Description and Priority

This feature will only be available to administrators. It will allow them to view all advisees assigned to particular advisors, as well as all students that have no advisors. All students should have an advisor, so this view will allow them to rectify the situation.

This feature is nonessential to the main functionality of this program and as such is a low priority.

3.6.2 Stimulus/Response Sequences

Selecting the 'Admin Advisor Manager' tab within the navigation menu will take the user to the advisor manager screen. Within this screen, there is a dropdown menu that allows the user to switch between every advisor at the university, as well as an 'unassigned' tab, which will be first chosen upon entering the screen. For each tab, the list of students that has that advisor will be displayed, along with major, graduation date, and email.

3.6.3 Functional Requirements

REQ-6.1: For each advisor, list each associated advisee along with relevant information

REQ-6.2: Prominently display students without an advisor so that situation can be rectified immediately

4. External Interface Requirements

User Interfaces

Majorizer's user interface follows several standard buttons and functions that permeate throughout the application in order to make it more intuitive and user friendly. First, Figure 1 shows the navigation bar at the top of the screen which is present across all of the major screens. This allows the user to access the home screen, log out, and access the side menu, from which they can access all other pages.

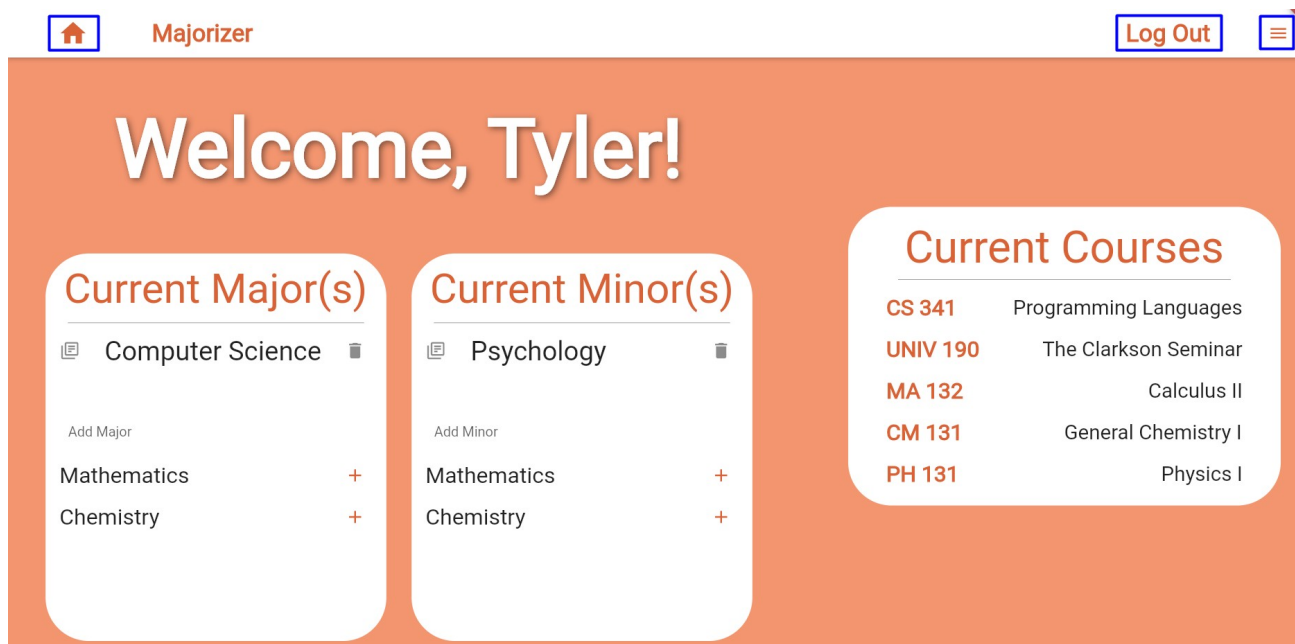
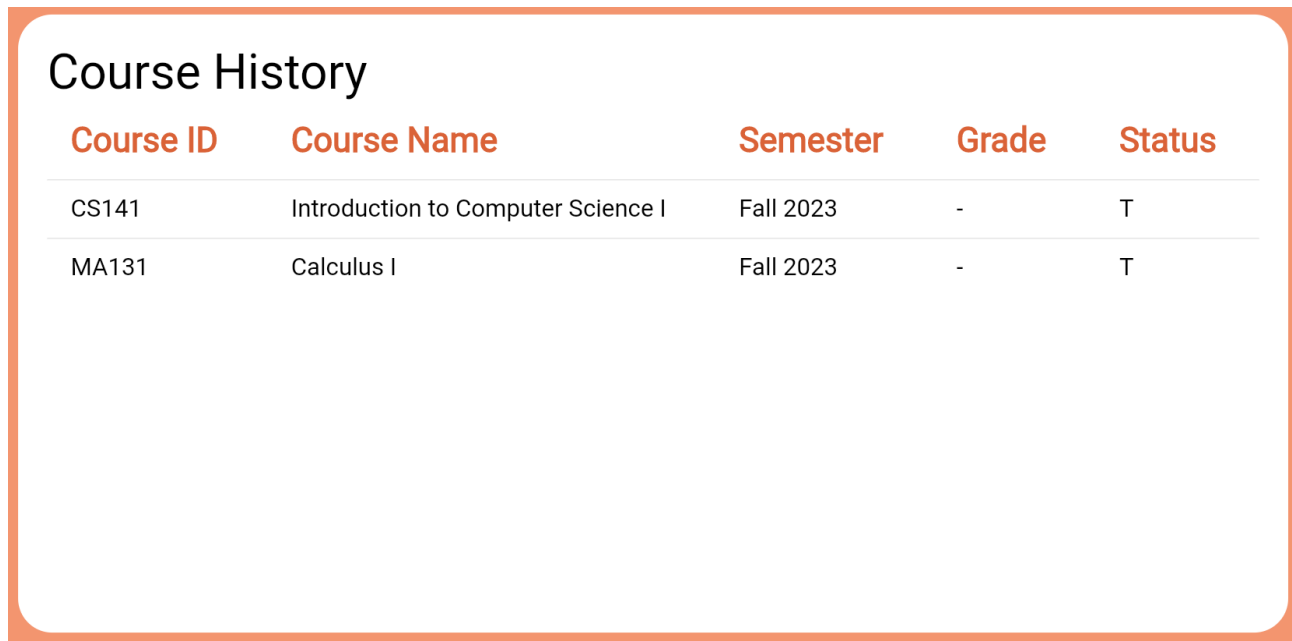


Figure 1: Sample Home Screen; General Navigation Tools Highlighted

Figure 2 shows a sample course history screen which is an example of the general table-based layout of many of the pages. Information pertaining to courses, students, and advisors is often shown in tables as this was determined to be a simple, user-friendly approach. Some of the tables, such as the course catalog, also provide drop-down menus for each row in the table to access more information (e.g., in that case, the course description is shown in the drop-down menu, which wouldn't fit anywhere else).

The image shows a web interface titled "Course History" with an orange border. It contains a table with five columns: Course ID, Course Name, Semester, Grade, and Status. The table lists two courses: CS141 (Introduction to Computer Science I) and MA131 (Calculus I), both from Fall 2023, with a grade of '-' and a status of 'T'.

Course ID	Course Name	Semester	Grade	Status
CS141	Introduction to Computer Science I	Fall 2023	-	T
MA131	Calculus I	Fall 2023	-	T

Figure 2: Sample Course History Screen

Hardware Interfaces

The software is designed to run on a Chrome web browser. As such, there is not much to note about the hardware needed for the program. It is worth noting that the software was designed for a desktop-sized resolution and as such may be harder to navigate/use on other devices, such as handheld phones.

Software Interfaces

The main database being used to store information on advisors, students, and courses is using MySQL 5.6.33. The server on which this database is stored uses Ubuntu 14.04.1. The database is managed by the Clarkson Master's in Data Analytics Program and can be accessed with verified credentials via mysql.clarksonmsda.org.

The backend of the software is using Python 3.9.7 with the following libraries:

- anytree 2.8.0
- asgiref 3.2.10
- Django 3.1
- djangorestframework 3.14.0
- django-cors-headers 3.11.0
- mysqlclient 2.1.1
- pytz 2022.7.1
- six 1.16.0
- sqlparse 0.4.3
- tzdata 2022.7

The nature of communication between the database and the backend is primarily read operations. For example, it pulls all data on courses to display in the Course Catalog feature. Some

information on course offerings is also needed for the schedule builder algorithm. This communication is done using raw SQL queries in Python. Django logins are also performed which interacts with the database indirectly and allows the software to get the user ID for the user which is making a given HTTP request. Some writing to the database is done, for example, when students add courses that they have previously taken; such writing is done using HTTP POST requests.

Communications Interfaces

HTTP is used as the primary communication protocol for transmitting data between the backend and frontend of the application. Information about students, advisors, and courses is formatted as JSON using Django and Flutter's native tools.

As mentioned previously, logins are handled using Google accounts, and the application is launched in a Chrome web browser, so there is communication regarding the information associated with the account. However, most of this is natively handled by Google.

5. Other Nonfunctional Requirements

Performance Requirements

It is especially pertinent to note the performance for the schedule builder feature. All other operations generally perform very quickly since they are not working on large amounts of data. The schedule builder algorithm is unfortunately not optimized for performance for this revision of Majorizer. For example, for a student majoring in computer science and minoring in mathematics, it might take 30 seconds to a minute to generate all possible course schedules for them. Future revisions would focus on optimizing the algorithm to only consider specific, more optimal schedules and store them in such a way that would reduce computational time, since this long wait time is not an optimal experience for the user. However, the current implementation is functional in the sense that it generates all theoretically possible schedules for the user which meet their degree requirements and only presents them with the shortest possible options (i.e. the least number of semesters to completion).

Safety Requirements

There is not much risk associated with the use of the Majorizer program. As Majorizer is in its early stages, it is possible that the security measures are not comprehensive enough to completely safeguard against all measures. Therefore, any data that is uploaded to Majorizer (which should only be course history and advisor information) may be subject to being accessed.

In addition, Majorizer should not be used to decide a course plan without the consultation of an academic advisor. Individual attention can help decide some things about a schedule that an algorithm cannot. For example, an advisor might recommend against overloading credits during a semester that has a famously difficult in-major course. This information is not something that Majorizer would have access to, so advisor consultation is still very important to take alongside the use of Majorizer.

Security Requirements

Majorizer stores basic information about users, such as their name, email address, and username, and user class (student or advisor), as well as information relevant to their use of the software, such as course history for students. It uses Google Firebase to handle user authentication, so passwords are not stored in the Majorizer database.

Software Quality Attributes

The most important attribute of Majorizer is that it is flexible. The schedule builder should be flexible enough to take any combination of majors/minors, unique semesters, and course histories and give users viable schedules if any exist. These can then be looked over with the assistance of an advisor. Following this, it is also important that Majorizer is testable, to ensure that any issues with any of these unique cases are caught before release.

Another important attribute of Majorizer is that it is updated to reflect any changes in course curricula as semesters progress.

Business Rules

For this software, there are three classes of users as laid out in an earlier section: students, advisors, and administrators. These correspond with the roles these individuals should hold at their university - students will login as a student, faculty with an advising role will log in as an advisor, and department chairs or other officials who oversee advising will login as an administrator. As noted in Appendix C, the following rules are not yet implemented in this code, but are the ideal standard.

The Advisor Manager screen should only be available to administrators.

The Student Manager screen should only be available to advisors.

The Schedule Builder and Course History screens should only be available to students.

The Advisor View screen should be visible to both students and advisors, although the exact nature of the screen will look different between these different user types.

6. Key Milestones

#	Milestone	Target Completion Date	Comments
1.	Requirements Interview	Jan. 30, 2023	Interview completed with customer Sean Banerjee to assess requirements of software
2.	Requirements Presentation	Feb. 14, 2023	Presented on chosen requirements to class
3.	Database design	Mar. 1, 2023	Structure of back-end database was completed
4.	Front-end Demonstration	Mar. 9, 2023	Demonstrated progress of front-end of application to class
6.	Project Completion	May 4, 2023	Presented final product to Sean Banerjee

8. Other Requirements

This project is not subject to any other overarching requirements due to it being a self-contained class project.

9. Requirement Change Management

For the scope of this project, requirements will not be changed. As such, there are no set guidelines on how to deal with this situation.

10. Restrictions, Limitations, and Constraints

This version of Majorizer only allows for input of a few majors and minors at Clarkson University, starting in Fall of 2023. A future version of Majorizer would allow for all major/minor combinations, and even allow for other universities to modify the program specifications so that it can be used by their students and advisors.

It is also worth noting that in any version of the program, building schedules that extend years into the future may not be entirely accurate, as universities can change course offerings or degree requirements. Therefore, it is important for users to revisit their course planning every university term, to confirm their plans and update them if required.

Appendix A: Glossary

Administrator: A user class that is intended for use by university administrators

Advisors: A user class that is intended for use by university faculty that advise students

Student: A user class that is intended for use by university students

User: Any person who uses the Majorizer program

User Class: A designator of whether a user is an administrator, advisor, or student

Appendix B: Analysis Models

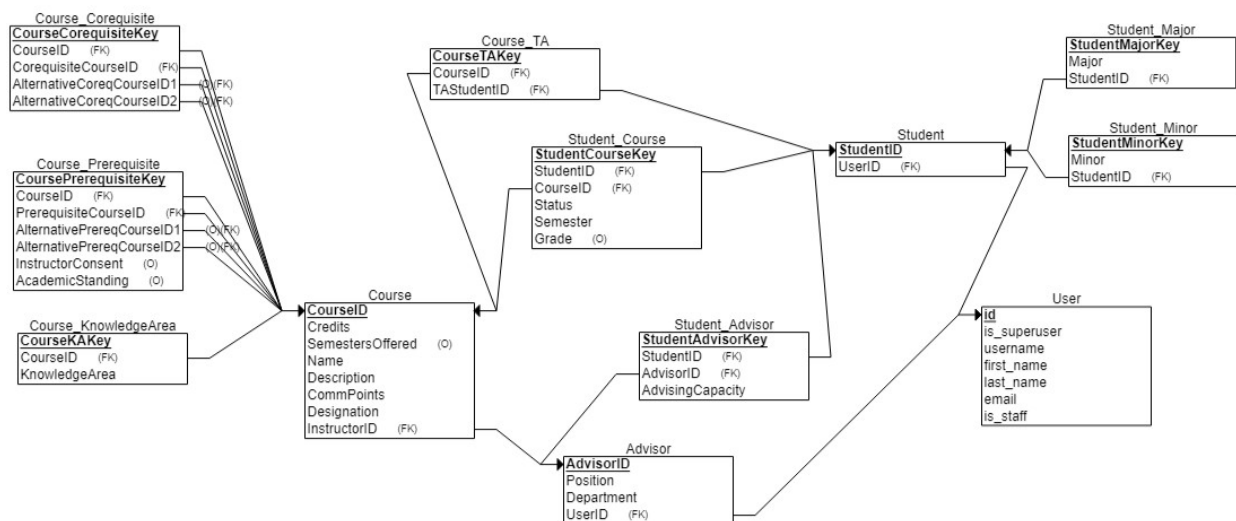


Figure 3: Relational Schema for Majorizer Database

Appendix C: Issues List

The primary major pending issue with this version of Majorizer is security. User authentication is done using email addresses on the backend rather than using Firebase token authentication. While we acknowledge that this is not a secure long-term solution, it got the system working in the time that we had.

Another issue that remains unresolved is making the schedule builder more customizable to each individual user's needs. Accounting for special circumstances such as study abroad, co-ops, or taking a gap year is the primary appeal of Majorizer, but we did not have time to fully implement such features in this revision. Future revisions would focus on further circumstances such as completion of a graduate degree or taking alternative courses.

Some other issues that have yet to be addressed are the separation of user classes upon signing up. The infrastructure exists to have users have one of the classes listed elsewhere in the document, but as of now, each page is accessible to each type of user, although each page may not be fully functional for non-intended users.

Finally, we did not have time to implement the course TA feature. This was a stretch goal that was intended to show advisors the list of courses they are currently teaching along with a list of student teaching assistants for each course. Unfortunately, we did not have time to complete this.