DSCI 551 Project Final Report:

Undergraduate Application Assistant

[Website](https://share.streamlit.io/prestonfong/scholarship-assistant/main/app.py?fbclid=IwAR0e_99w8BK0gDXqSer3_14z9UUBNVmL22cAjUaviHSHW2l0_cLOMOonCho)

Preston Fong

Andy Xiang

April 24, 2022

## Project Title

[Undergraduate Application Assistant](https://share.streamlit.io/prestonfong/scholarship-assistant/main/app.py)

[Project Files](https://drive.google.com/drive/folders/1Htvz-uqj3C61zd-MmsQ4PsbDDEkoHKHe?usp=sharing)

## Topic

This app will serve as a resource by providing college application information, deadlines, scholarship details, and other vital information for prospective undergraduate students.

Students can input their personal and academic information to see their recommended colleges as well as their eligible scholarships. We have also built in a reminder feature that can allow users to set phone or email reminders of upcoming deadlines. The purpose of our project and web application is so that we can help underserved high school applicants better be informed and ready to apply to colleges in California.

## Motivation

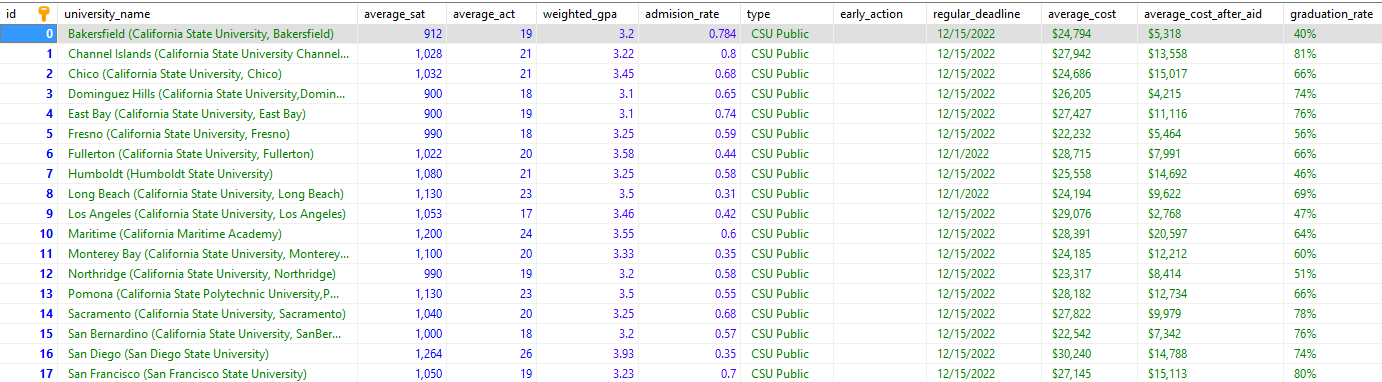
The US has almost 500 students for every guidance counselor. This means that the majority of students are not receiving the necessary help they need to understand the college application process. In fact, the lack of guidance for college admissions disproportionately affects low-income, first-generation, and minority students. Our app will be able to help underserved students apply for colleges by providing the necessary information, resources, and guidance.

As two students who recently graduated from California universities, we understand the challenges of applying and navigating the college application process. The information is overwhelming and many underprivileged students do not have the resources to know which colleges to apply to. Thus, we decided to build a web application that can consolidate all that information and help with the undergraduate application process.

## Data Collection

1. Freshman Admission Data

We were able to collect admission data on over 45 private and public universities in California. We developed a web scraping program using the python library Selenium to google search important data elements such as average GPA, average cost, admit rate, etc. Selenium was only able to scrape roughly 85% of the data we needed, and we decided to do a manual search on the missing values. Thus, we have a complete and well-formatted freshmen admission dataset in California.

****

**Image 1:** Freshman Admission Database in Amazon RDS

1. Scholarship Data

We mostly did manual searches to find legitimate scholarship links and data as most scholarship websites require a subscription or membership. This process was longer than expected, but we believed this data would be extremely valuable and useful to our users. We have collected over 40 legitimate scholarship links that are tailored specifically to underserved and prospective undergraduate students.

****

**Image 2:** Scholarship Database in Amazon RDS

## Data Cleaning

After gathering our data via Selenium or manual searches, we formatted our data using Excel. Most of our data cleaning and data formatting were done in excel as there were less than 200 rows collectively. Since our data was not extraordinarily large, we decided that cleaning our data on python was not necessary. To ensure that our data would be displayed properly on the UI, we made sure that the data was formatted correctly amongst each cell in every column. We also made sure to fill in the missing values, using the most reasonable assumption, wherever possible as it would be better for our users’ experience. Once our data tables are cleaned and complete, we would upload the data into our cloud database using Amazon RDS.

## Architecture

1. **User Interface and Streamlit**

To develop our UI we decided to use streamlit to build and enhance our user interface. For any user to start running our program, they would simply download our entire project file and then run “streamlit run app.py” on the terminal. A new window should pop up and the user can access all the forms, databases, and features of our web application. We have also decided to public host our [website](https://share.streamlit.io/prestonfong/scholarship-assistant/main/app.py?fbclid=IwAR0e_99w8BK0gDXqSer3_14z9UUBNVmL22cAjUaviHSHW2l0_cLOMOonCho).

The home page of our web application displays our main feature which is recommending the colleges given your academic and professional merits. After the user finds their recommended colleges, they can go on our navigation tab and find the scholarships that they would be eligible for. Lastly once the user has found their recommended college and scholarships, the user can set reminders for upcoming scholarship and college application deadlines. The user has the ability to set the reminder date as well as remove the reminder.

1. **Amazon RDS Cloud Database**

Using Amazon RDS, we created a cloud mysql database that can be easily accessed in multiple ways. Our primary way of connecting to the database is through pymysql and MariaDB. MariaDB was essential in helping troubleshoot the database and quickly fix problems. We utilized it to run queries and inspect inserted data. PyMySQL allowed us to automate the data insertion and our application to interact with the database. We are able to query our database of colleges, scholarships, and reminders as well as insert new reminders.

1. **Twilio API (Phone numbers)**

To build our reminder feature, we used the Twilio API, which is a cheap paid client that allows users to automate text messages and phone calls. For this use case, we used the API to send users text message reminders. We first purchased a local phone number and connected our program to the Twilio client. This code can be run separately as it is only needed to query the database once a day to send text messages. To see the code, please visit messages.py

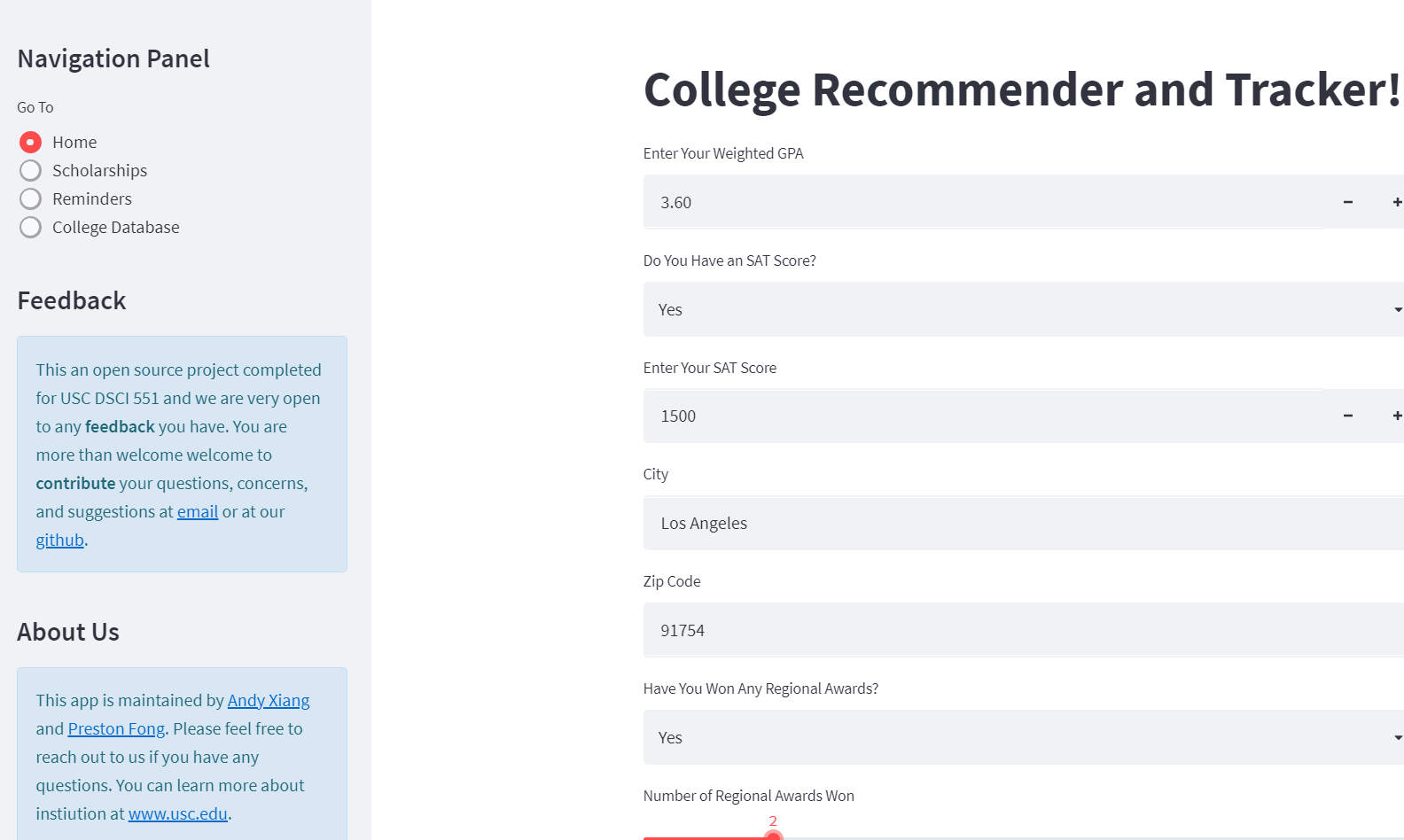
1. **Yagmail (Sends emails through a Gmail client)**

To allow users to be reminded via email, we used Yagmail which is a free client that allows us to send users email reminders when an upcoming deadline is approaching. We created a new email called: [dsci551reminders@gmail.com](mailto:dsci551reminders@gmail.com) that will send email reminders to the users if they choose to opt-in. To see the code, please visit messages.py

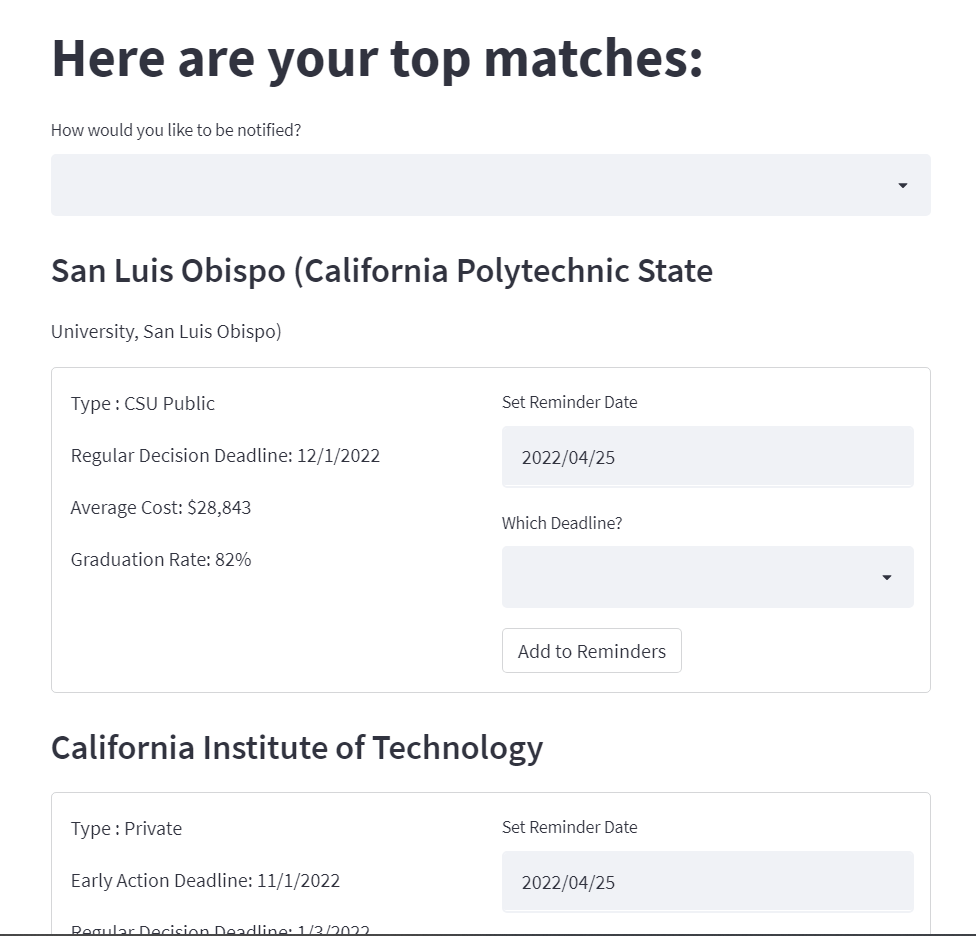
## Components & Main Functions

1. **College Predictor Form**

The primary function and feature of this web application is so that high school students can input their academic information and our program will provide them with the recommended universities that they should apply to. We have collected data on over 45 California private and public universities and developed an algorithm that considers your GPA, SAT, ACT, and merit awards when providing your college recommendation. We have considered adding in elements such as race, gender, first-generation status, or socioeconomic statuses, but we did not have sufficient data to develop an algorithm that can consider these factors. Once the algorithm has matched the student with schools, the student will be able to set a customized reminder for each school.



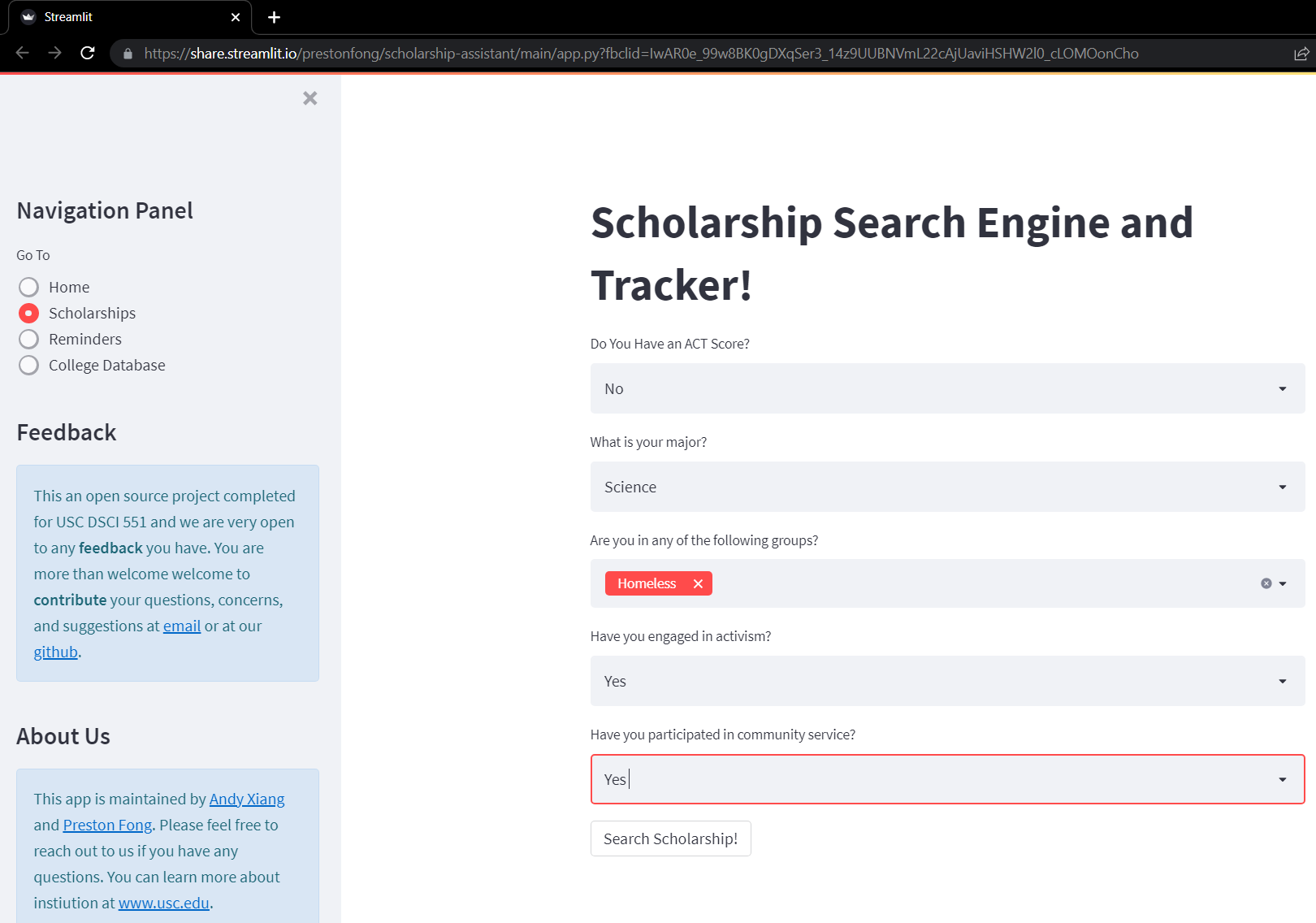
**Image 3:** Home page and college recommendation form



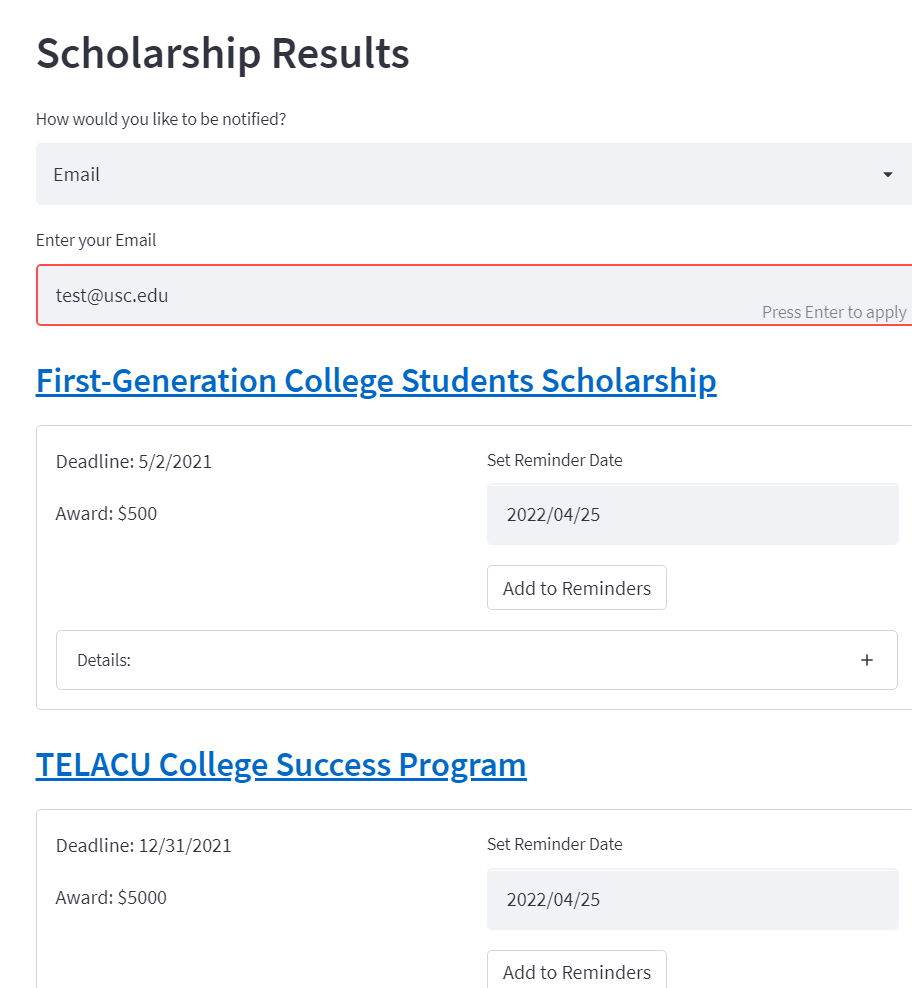
**Image 4:** Display of college recommendation results page

1. **Scholarship Form & Database**

We have also built a form for users to determine which scholarships they would be eligible to apply for. Users would have to quickly provide some personal and demographic information to see which scholarship they are eligible for. The user would be redirected to the results page and see all the scholarships given their eligibility criteria. Similarly, they will be able to set reminders here.



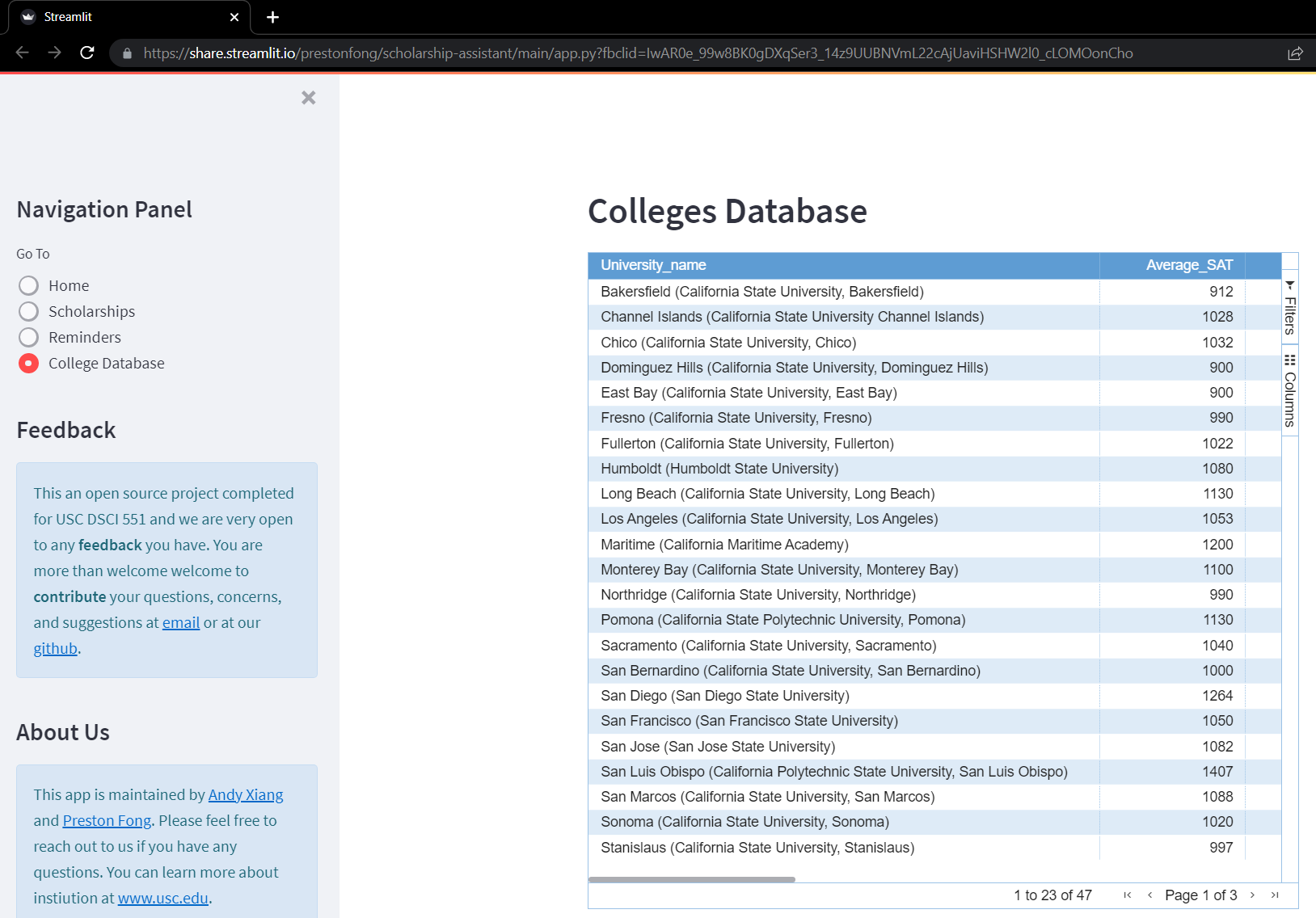
**Image 5:** Scholarship form



**Image 6:** Scholarship results page

1. **College Database**

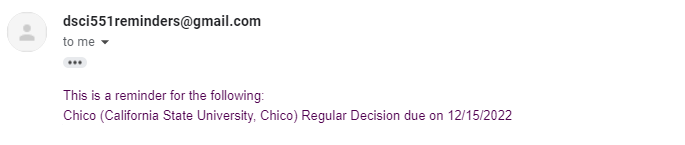
One of the by-products of creating a college predictor form was developing an entirely well-formatted database on university freshmen admission data. Users can view this table on our webpage and make informed decisions by themselves. We were able to collect critical data features for 2022 such as Early Action deadlines, average GPA, average cost after aid, etc. Users would then be able to easier compare certain features of colleges against colleges more visually. We believe this database would be very useful for guidance counselors, prospective undergraduate students, and education administrators.



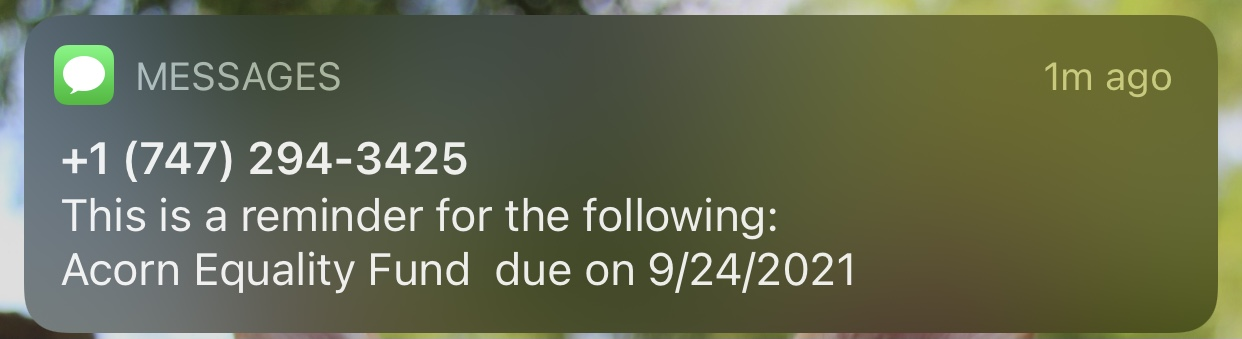
**Image 7:** Full college database

1. **Email & Text Reminders**

The web application is able to send push notifications and email reminders to students when an upcoming college or scholarship deadline is approaching. The user has the ability to set the reminder date as well as remove the reminder. We have built a friendly UI feature that allows users to see all the existing reminders and remove any additional reminders.

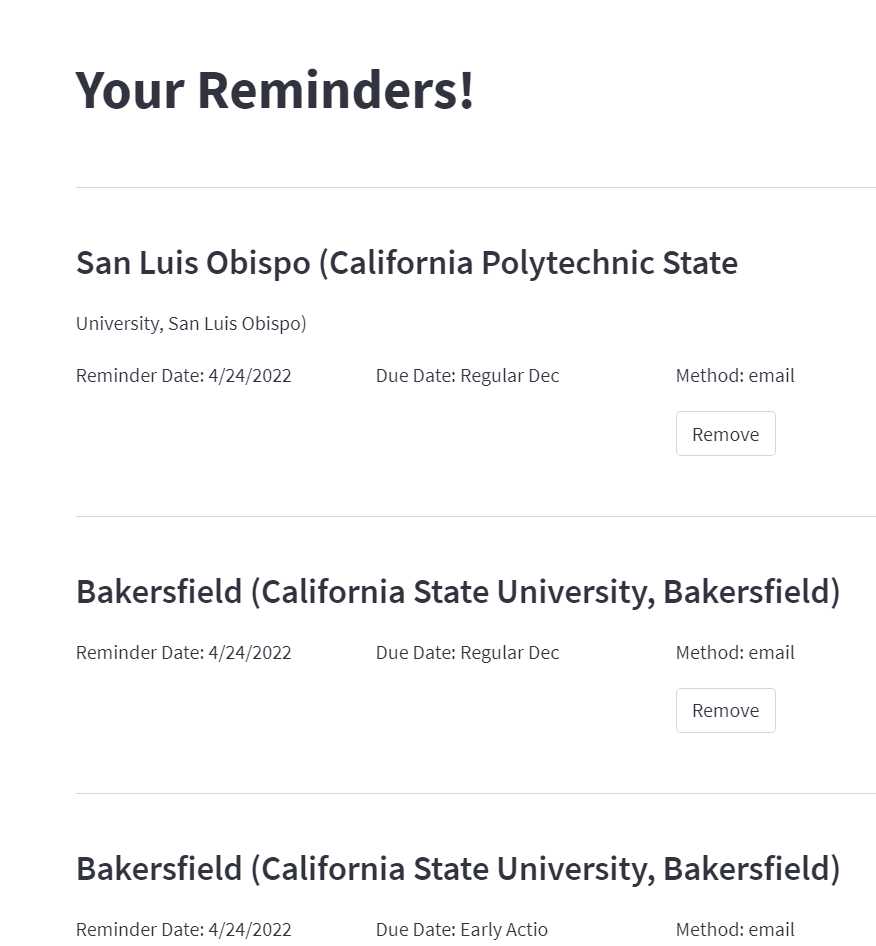


**Image 8.1:** User email reminder



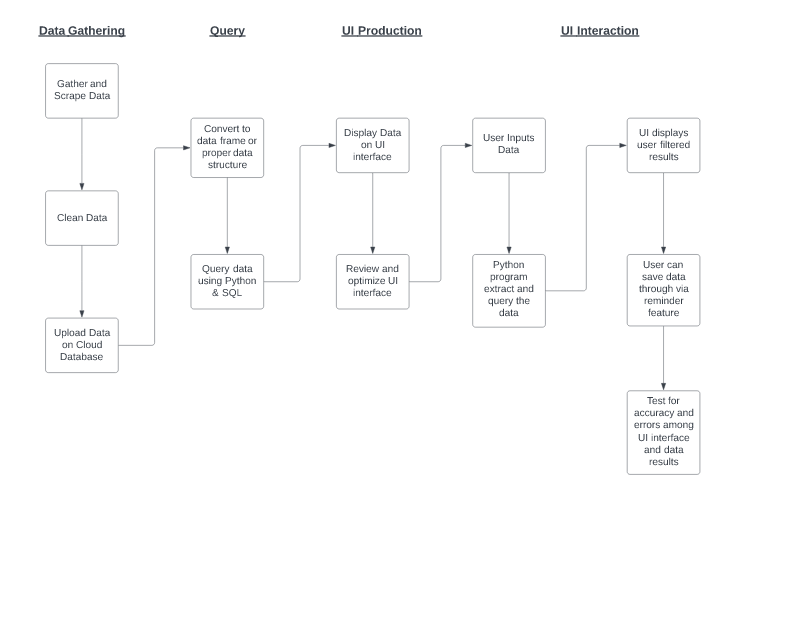
**Image 8.2:** User text message reminder

| **Image 8.3:** User chooses to opt in reminders via email |
| --- |

****

**Image 9:** User can see current reminders opted-in and remove any reminders

## Data Flow

**We would first gather the data through web-scraping or manual searching. After cleaning the data, we would upload our data onto Amazon RDS, which is our cloud database. We used Python and SQL to extract the data from the cloud database and would convert it to a data frame or the proper data structure so that we properly display the data in a user-friendly format. The user would then input their personal and academic information and our program would use the inputted data to filter our cloud databases and display their unique results. We have tested our data flow and programs for bugs and have tried to improve the user experience. **

## Team Members and Responsibility

Andy Xiang:

1. Gathered and cleaned the college admission database
2. Drafted the code using streamlit for user interface and experience
3. Tested the UI code for any corner cases and errors
4. Drafted the midterm and final report
5. Edited and finalized video project

Preston Fong

1. Gathered and cleaned the scholarship database
2. Built the Amazon RDS Cloud database and stored all data
3. Built the front-end and back-end reminders feature
4. Finalized and enhanced the code using streamlit for user interface and experience
5. Tested the UI code for any corner cases and errors

## Reflections on Learning Experience

Andy Xiang:

Overall I found this project to be very useful and exciting. I have really had the opportunity to work with UI & UX and streamlit was a great library and experience for us to get started. I also found streamlit to be extremely easy to use for someone with limited HTML and CSS experience. I also really enjoyed making a web application as this allowed us to showcase our work as we believe that this feature can be useful to a wide range of users.

The only change I would make if I were to do this project again would be to build a machine learning algorithm that can predict the user’s colleges, which can provide the likelihood the user will be accepted (assuming that data can be found). Overall I had an enjoyable time learning new technologies such as building cloud databases and user interfaces. I feel that these are invaluable skills and experiences for my future career.

Preston Fong:

At first, I was fairly anxious about building a web application for a data science project. My limited experience with HTML and CSS was not pleasant. However, working with streamlit made things much more enjoyable, and I am excited about our finished product. I feel confident that in the future I would be able to draft and deploy a web application for a data project quickly using streamlit.

Trying to source and extract free college data was extremely time-consuming This illustrates how valuable clean data can be. I hope in the future I will be able to develop more projects like this that are invaluable to many people across the world. I truly believe that we were able to provide a useful resource for high school students, and I really enjoy the entire process of this project.

## Challenges Faced

1. Streamlit Limitations

One of the biggest challenges was customizing the UI platform. Since we used Streamlit which automates the majority of user interface and experiences, we were limited in the option of customizing certain features. For instance, we were unable to change the display format of certain texts and dates. We were also unable to modify the spaces or add line breaks to improve the user interface. Nonetheless, we believe UI still appears to be very friendly and easy to use.

1. Data Collecting

One challenge we encounter is the process of collecting data. There is a vast amount of data on the internet and collecting, cleaning, and then storing the data poses a challenge as data can vary very differently. We have noticed that a lot of datasets have enormous amounts of missing values and unnecessary information. It has taken us a long time to identify the most important information in order to judge the quality of our datasets. To build our scholarship database, we did a lot of searching as there we could not solely rely on one website to gather all our scholarship data. This issue was also common when we were trying to build a college admission dataset.

1. Standardizing Data

Another challenge that we faced was the lack of standardization in these datasets. In order to merge information from different datasets, we needed a unique ID so that we can join this data. This is why we created the college\_ids table. This table will contain unique information from the college that is easily matched from various datasets.