SW Engineering CSC648/848 Fall 2021

Project Title: Gator Learn

Team Number: 04

Names of Students:

Manali Seth (Team Lead) (Email: mseth@mail.sfsu.edu)
Htet Soe (Front-end Lead)
Aarshil Patel (Back-end Lead)
Aditya Mohan (Github Master)
Seela Pant
Christian Samatra
William Yu

Milestone: 2

Date: 10/28/2021

History Table:

Date	Revision
10/28/2021	Created as the first version
11/10/2021	Revised as per the feedback

Table of Contents

1. **Executive Summary**

Prior to the pandemic, students could easily seek help from resources provided by the school on campus. However, this has totally changed, and students are struggling to find these resources remotely. The pandemic has changed the way students approached their learning. Remote learning has made it difficult for students to seek help with their academic assignments as our community practices social distancing. With that in mind, the idea behind Gator Learn came to fruition. The basic idea behind this application is to help Gator students find the necessary help for their assignments remotely to enhance their remote learning experience.

During this time, we understand that our fellow Gators are under pressure more than ever with the drastic changes in how we approach our learning sessions. We are trying to alleviate some of the academic pressure by creating Gator Learn. Gator Learn is a web application that will provide a way for San Francisco State University students to easily find tutors that will help them with their assignments, tailored specifically to our offered Gator courses. With an easily accessible menu to find your particular course, professor, or assignment, our web app outshines other competing academic assistance sites as a Gator student. At Gator Learn, the tutors can be former Gator alumni, Teaching Assistants, or professors who have experienced what it is like to be part of the San Francisco State University community. They will be able to relate with the students seeking help and be able to provide the necessary assistance for each student. In short, Gator Learn will allow San Francisco State University students to have access to qualified tutors for all their assignments.

Gator Learn will be produced by a very diverse Team 04 group from San Francisco State University. Each member of the group has come from different backgrounds that has one goal in mind: produce a product that will benefit our fellow students at San Francisco State University and help them achieve their academic goals. As a team, we believe we will produce a product that will compete with other similar apps that are available online. Our team is also composed of San Francisco State University students and understands what our peers are struggling with as we struggle with it ourselves. With the team member’s expertise and experiences, the Gator Learn will be a product that will not only help current San Francisco State University but also future students. And, as we improve our product, hopefully it can expand to help other students from other universities.

2. Main data items and entities

1. Major: A category of classes designated to a specific major/topic (ex. CSC in CSC-648).

- Major ID : Primary key for Major table
- Major Name: List of all majors available in SFSU University

2. Course: A course at SFSU belonging to a major.

- Course Number: Primary Key for course table. The course number (ex. 648).
- Availability: Shows whether the course is available this semester at SFSU.
- Course Name: The course name (ex. CSC - Computer Science).

3. Catalog: A relationship table which has a listing of all courses and majors at SFSU, currently available or not. Used for creating a database of all SFSU majors and courses on the website.

- Major ID: Primary Key and Foreign Key referencing Major ID from Major table.
- Course Number: Primary Key and Foreign Key referencing Course Number from Course table.
- Requirement: States if the course is required for the major.

4. Unregistered User: A person who has simply found the website and is using it without signing up.

- Session ID: Primary Key for Unregistered User table. A session id will be allocated to any anonymous user just browsing through the website.

5. Registered User: Describes user related information and also the status whether the user is unverified by admin or registered after approval or rejected by admin.

- User ID: Primary key for student or tutor.
- Session ID: Foreign key referencing Session ID from Unregistered User table.
- Major ID: Foreign key referencing Major ID from Major table.
- First Name: Specifies First Name of the user.
- Last Name: Specifies Last Name of the user.
- Date of Birth: Specifies birth date of the user.
- Gender: Specifies gender of the user.
- Email: The email that the user is using.
- Password: The account's password that is used for logging in. Kept confidential from other users, excluding administrative permissions.
- Avatar (Optional): Users can upload their image/avatar.
- Description (Optional): Some provided information about the user.

    - (Student) Mentions the degree the student is studying. (Ex: Grad / Undergrad)
    - (Tutor) Basic information about their tutoring and their qualifications.
- Current Semester: Mentions the current semester the student is studying. (Ex: Freshman / Sophomore / Junior / Senior / Graduate)
- Role: Describes the role of the user whether he is a student or a tutor.
- Course Taught: Specifies the course taught by the tutor so that the student can look into it and book an appointment for the same.

6. Tutoring Session: A listing for tutoring in qualified courses. Created by tutors and can be applied to by students. The following local details of the tutoring session can be seen by users browsing and are defined by the tutor

- Tutoring Session ID: Primary key for the tutoring session table.
- Tutor ID: Foreign key referencing tutor ID from Tutor table.
- Major ID: Foreign key referencing major ID from Major table.
- Course Number: Foreign key referencing course number ID from Course table.
- Status: Displays whether a particular session is still available or has filled to capacity.
- Timeslot: The time slot that the tutoring session is available.
- Date: The date that the tutoring session is available.
- Location: The location of the tutoring session is, also shows whether it is in-person or online.
- Review: An area for students to leave feedback of the tutoring session that they received.

7. Student Session: A relationship table between Student and Tutoring Session.

- SFSU ID: Primary Key and a Foreign key referencing SFSU ID from Student table.
- Tutoring Session ID: Primary Key and a Foreign Key referencing Tutoring Session ID from Tutoring Session table.

8. Profile page: A user-customizable page. A student or tutor can use this page to provide more/ edit information about themselves.

- User ID: Primary and Foreign key showing who’s profile page this is.
- Contact information.
- (Students) Course and major information.
- (Tutors) Credentials and Description.
- Other miscellaneous information.
- Comment Section: Other users may leave comments, or reviews in the case of tutors.

9. Administrator: Has full system control of the website. Administrators can approve tutors and student profiles. Furthermore, they can also monitor and remove tutoring session listings and profile pages for breaking website guidelines.

10. Inbox: A place for communication within the tutoring website.

- Reply: A user can respond to a message with their own.
- Read message: A receiving user accesses the contents of the message that was sent to them.
- Delete message: A receiving user deletes a sent message to them from their inbox.
- Status: Shows whether the message is read or unread.

11. Message: A message sent from one user to another on the site. Defined as a form by the following local details:

- Message ID: Primary key for message in the database.
- Tutor ID: Foreign Key referencing tutor ID from Tutor table.
- SFSU ID: Foreign Key referencing student SFSU ID from Student table
- Sender: The user who sent the message. Defaults to the user that is signed in.
- Receiver: The user who is receiving the message. Defaults to the user that the sender is trying to message.
- Date: The date that the message was sent.
- Time: The time that the message was sent.
- Subject: The title/header of the message.
- Contents: The contents of the message.

3. Functional Requirements (Prioritized)

Priority 1:

Unregistered user

1. Unregistered user shall be able to search for tutors by choosing major and courses in the search bar.
2. Unregistered user shall be able to browse courses on the website.
3. Unregistered user shall be able to register
4. Unregistered user shall be prompted to create an account (lazy reg.)
5. Unregistered user shall be able to view the tutors available.

Registered user

6. Registered user shall inherit all the functions from unregistered users.
7. Registered user shall be able to login to the website.
8. Registered user shall be able to apply for tutor posting.
9. Registered user shall be able to send message to tutors.

Priority 2:

Unregistered user

10. Unregistered user shall be able to access the contact page of the website.
11. Unregistered user shall be able to see course details and requirements.
12. Unregistered user shall be able to browse courses based on filters on the website.

Registered user

13. Registered user shall be able to logout.
14. Registered user shall be able to have drop-in sessions with tutors.
15. Registered user shall be able to provide feedback.

Admin

16. Admin shall be able to approve or reject tutor posting before they go live.
17. Admin shall be able to provide online support for technical issues.
18. Admin shall be able to delete accounts permanently.
19. Admin shall be able to modify majors and courses available at SFSU.

Priority 3:

Unregistered user:

20. Registered user shall be able to edit preferences and profile.
21. Registered user shall be able to access the FAQ of website.
22. Registered user shall be able to choose the mode of appointment through the portal (one-on-one or drop-in).

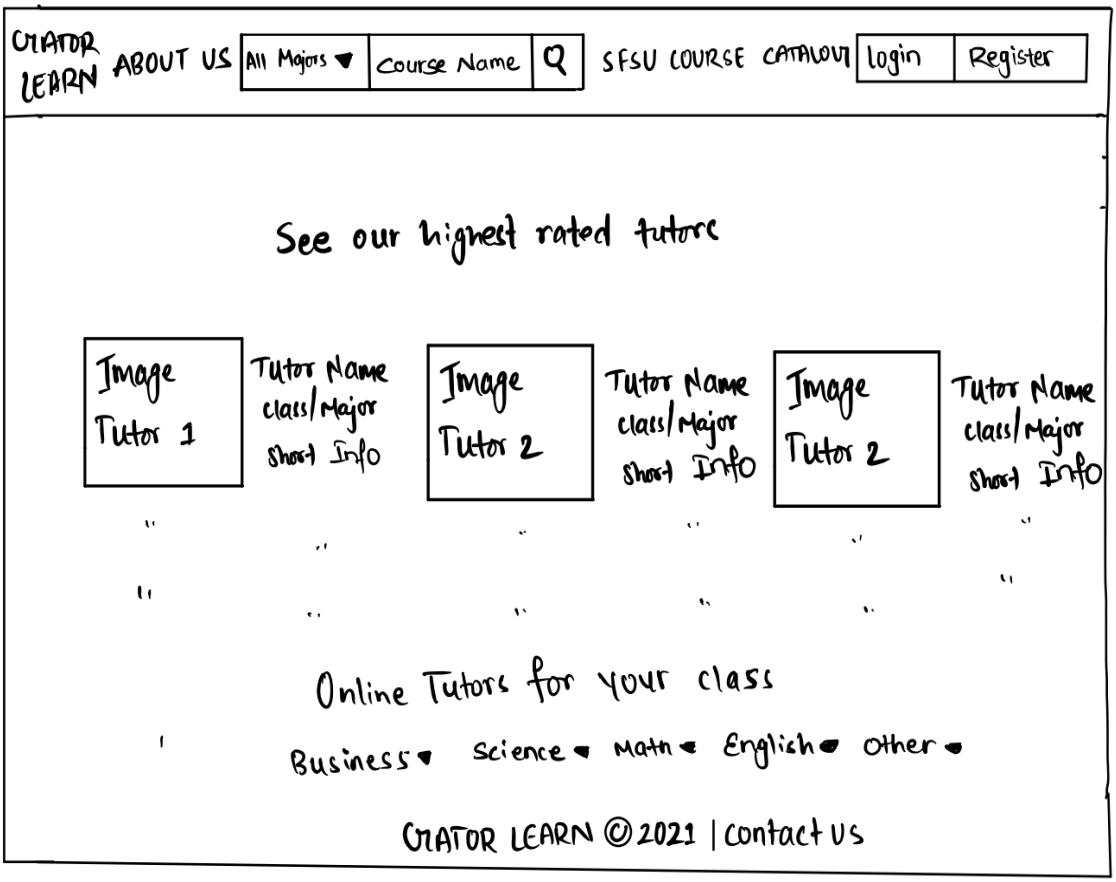
Admin:

23. Admin shall be able to reset user passwords.

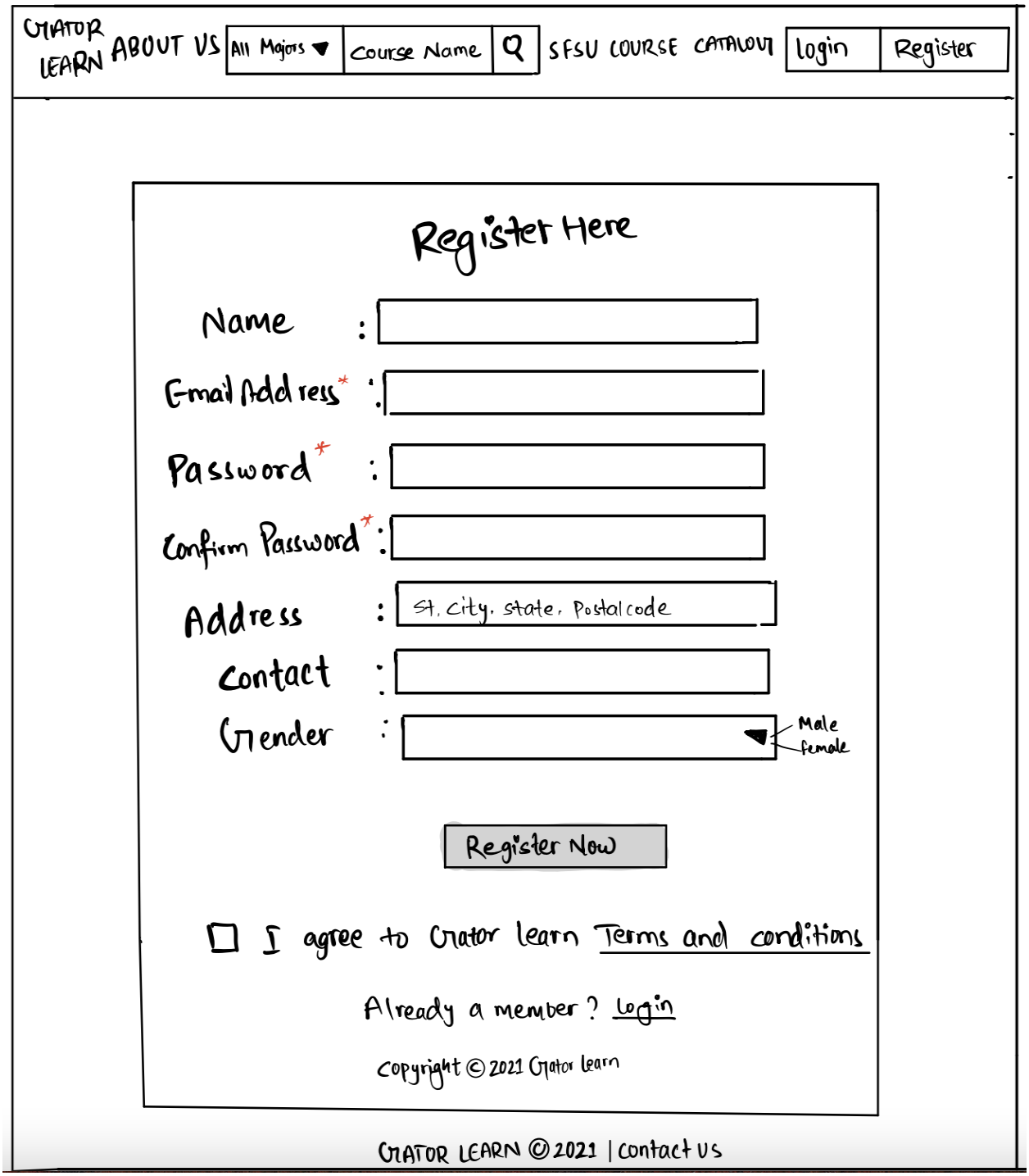
4. UI Storyboards

1. Signing up as Internal or External tutor

John is a former student at SF State University. He used to be an excellent student during his undergrad journey. As he follows the SFSU page on Twitter, he came to know about the tutoring website. He knows very well about the stressful moments for students in their class. Therefore, he wants to sign up as a tutor in order to help the students to succeed in their class. He was familiar with the online applications. So, he entered the requested tutoring info and then was prompted to sign up.
On the other hand, Janet is a 50 years old business woman. She completed her grad and undergrad decades ago from UC Berkeley. She is great at communicating and a very patient woman. She always wished and wanted to connect with students to help in their studies. She was able to easily sign up on our website because the website was very self-explanatory and easy to go through the whole process.



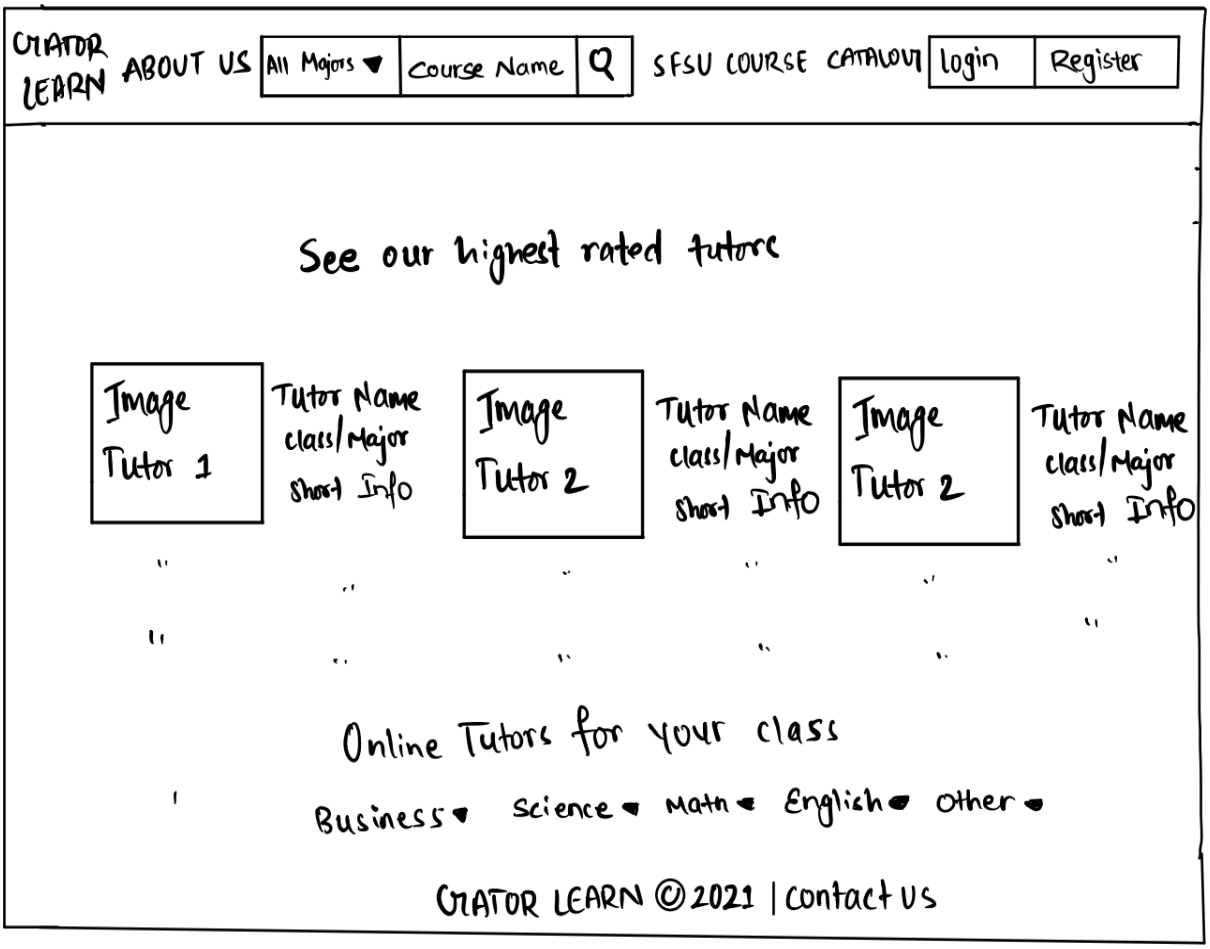
Home Page



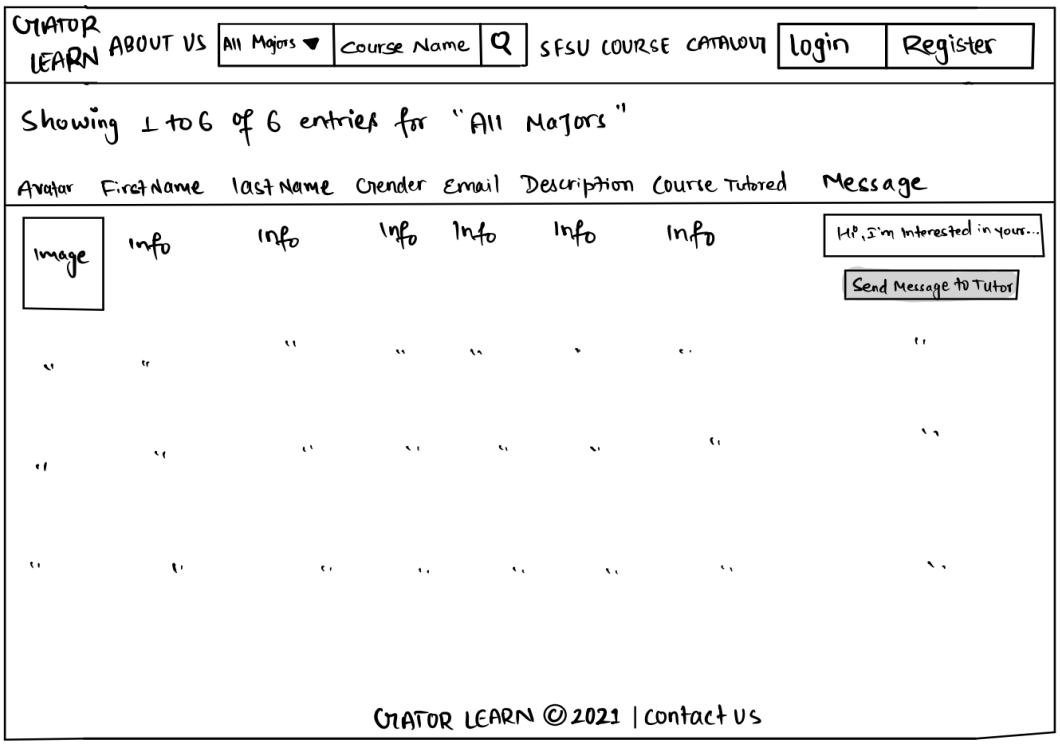
Registration Page

2. Browsing as a non-registered user

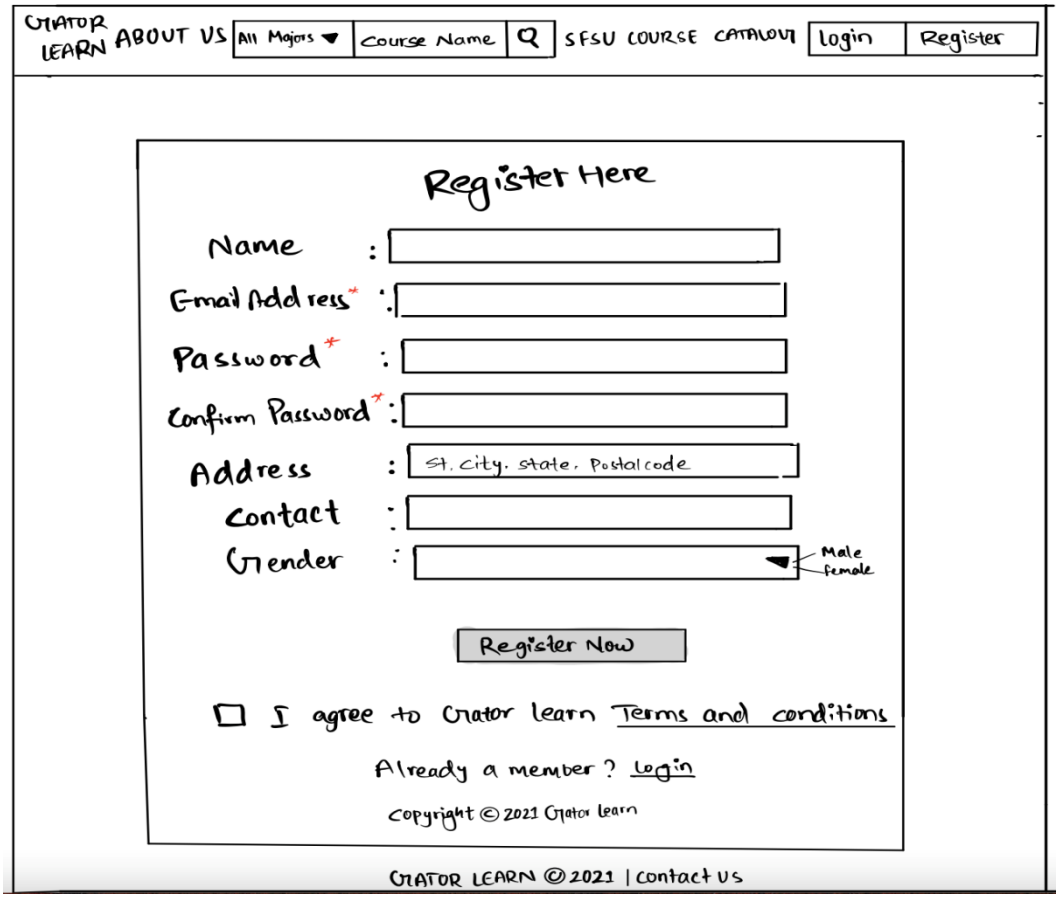
Jane recently transferred to SF State University. Being a junior is always a stressful year during college. She is an excellent student and very organized with her study life. She had a good GPA when she transferred with a Computer Science major. Because she is new, she has not made good friends yet. However, she is starting to need one whom she can work together with in her class. She does not want to mess up her grade as well. Fortunately, she heard about the "Gator Learn" from one of her friends in the class. She was so happy to go through the website and get help from a tutor. Jane is very good at self-navigating the websites. So she went to a website and started searching for tutors by major or course. She also found one according to her course and wants to schedule tutoring sessions with that respective tutor to help herself stay accountable. Before being able to schedule an appointment, Jane is prompted to register or login. Since Jane is a non-registered user she goes through the registration process.



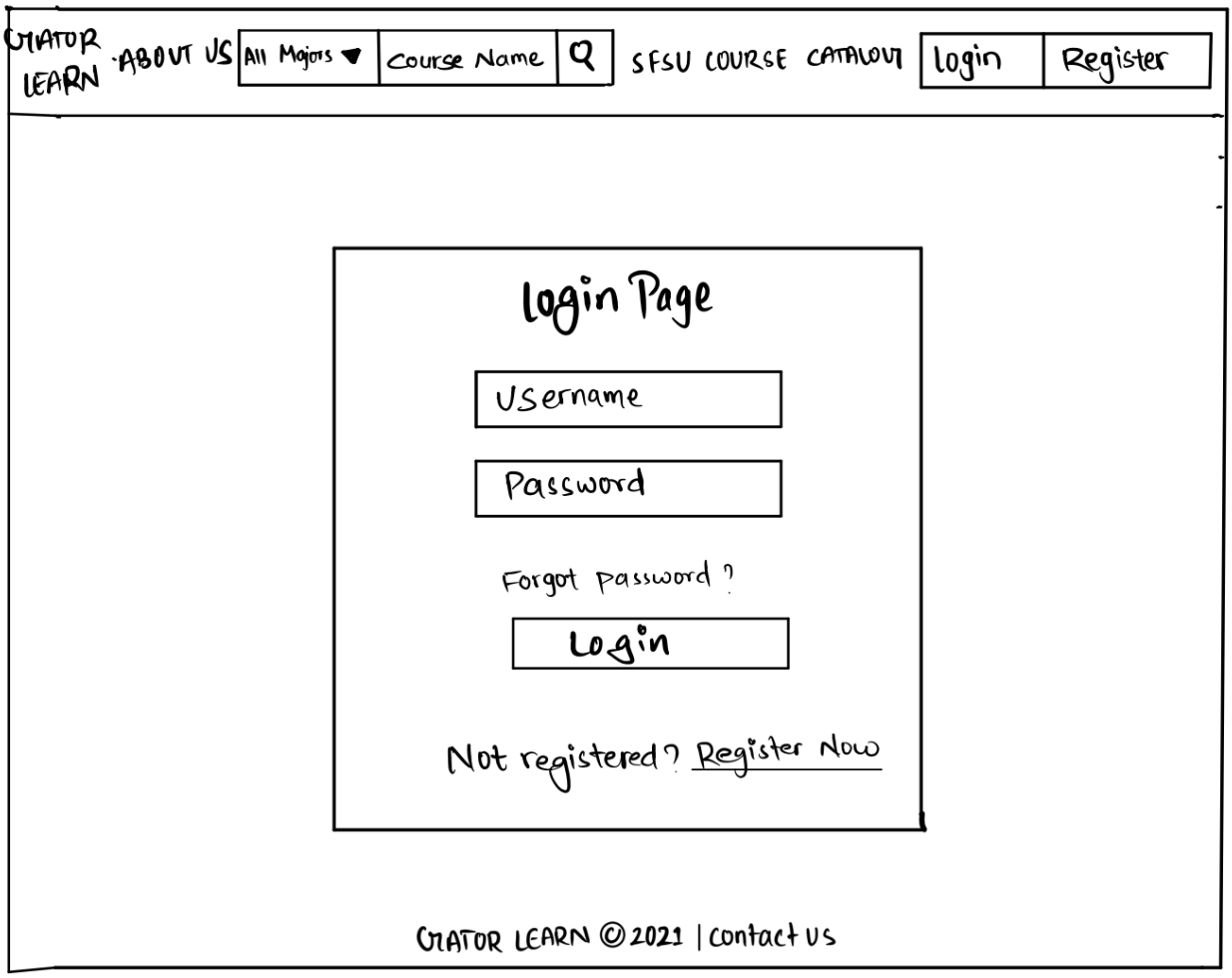
Home Page



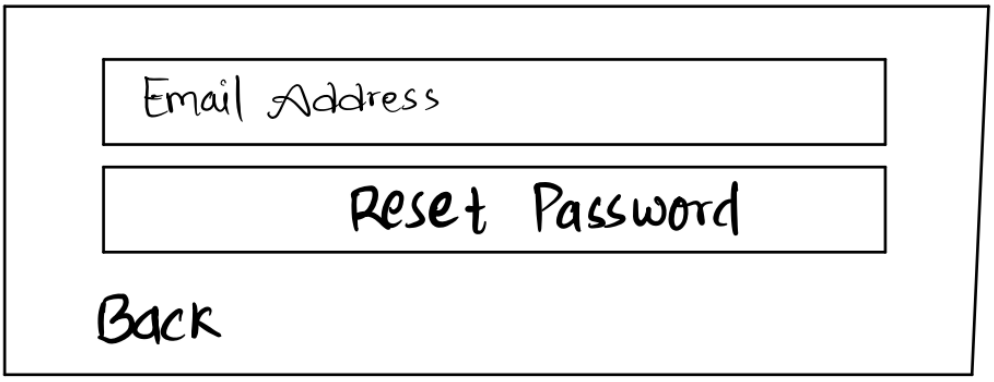
Search Results (After clicking on search button from navigation bar)



Registration Page (Lazy Registration) while trying to send message to tutor to book an appointment



Login Page (Sign-in so as to schedule an appointment or send message to tutor)

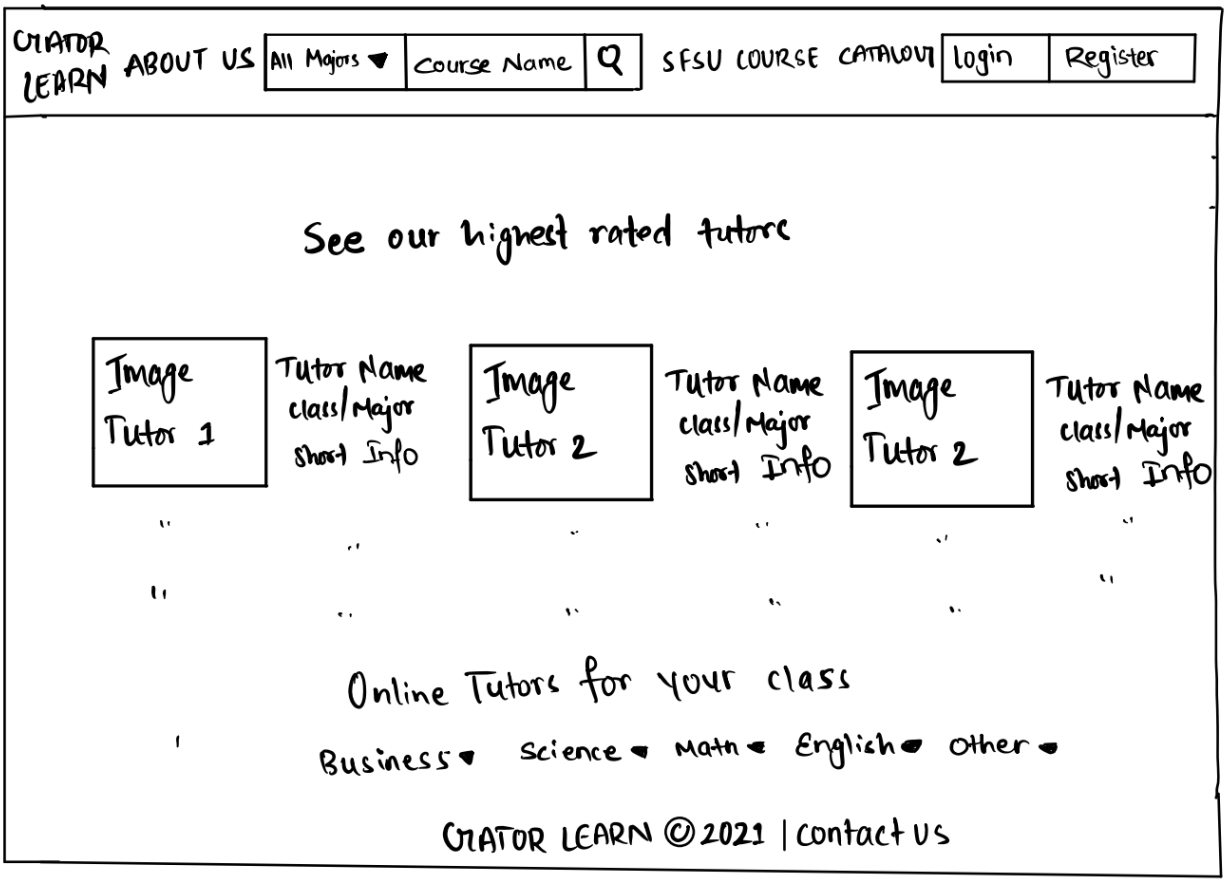


Forgot Password Page (Incase user doesn't remember the password)

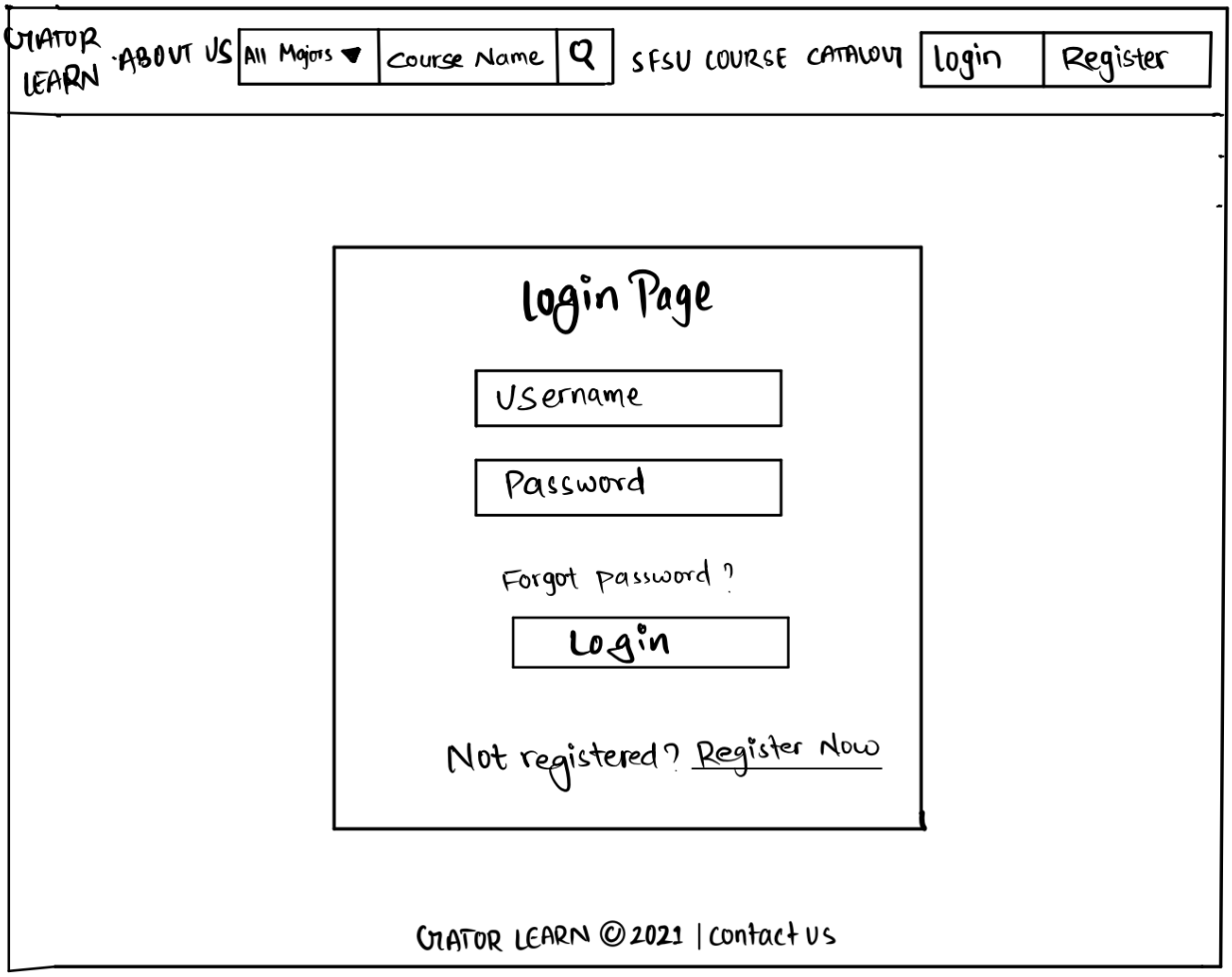
3. Scheduling appointment as a registered user

Josh is a senior student with an English major at SF State University. He is very busy with work and studying together. Because of this he is having a hard time managing time for his study. In the middle of the semester, he feels like he needs help with his math class. While he was working, he only had a phone with him, so he used his phone to navigate through our website. Since he has already registered on our website, he can sign in and start looking for the tutor he needs help with. He logs in, searches for the class, and checks for the availability of the math tutor. After checking the available spots, he schedules an appointment for him.

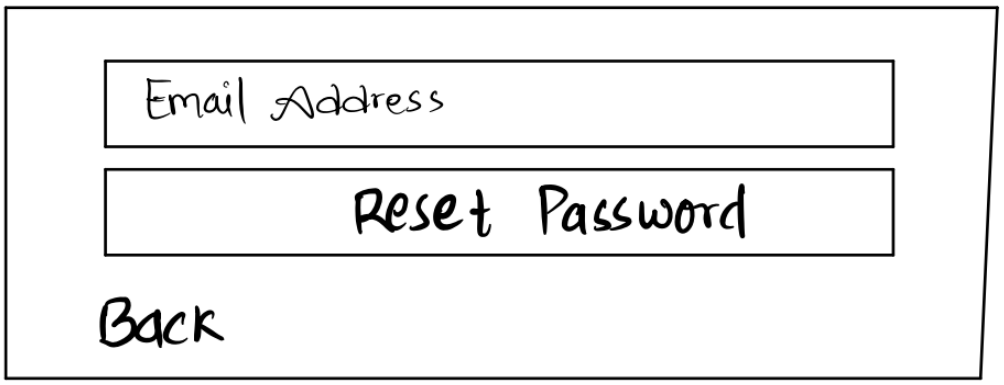
Available times are seen, otherwise not.



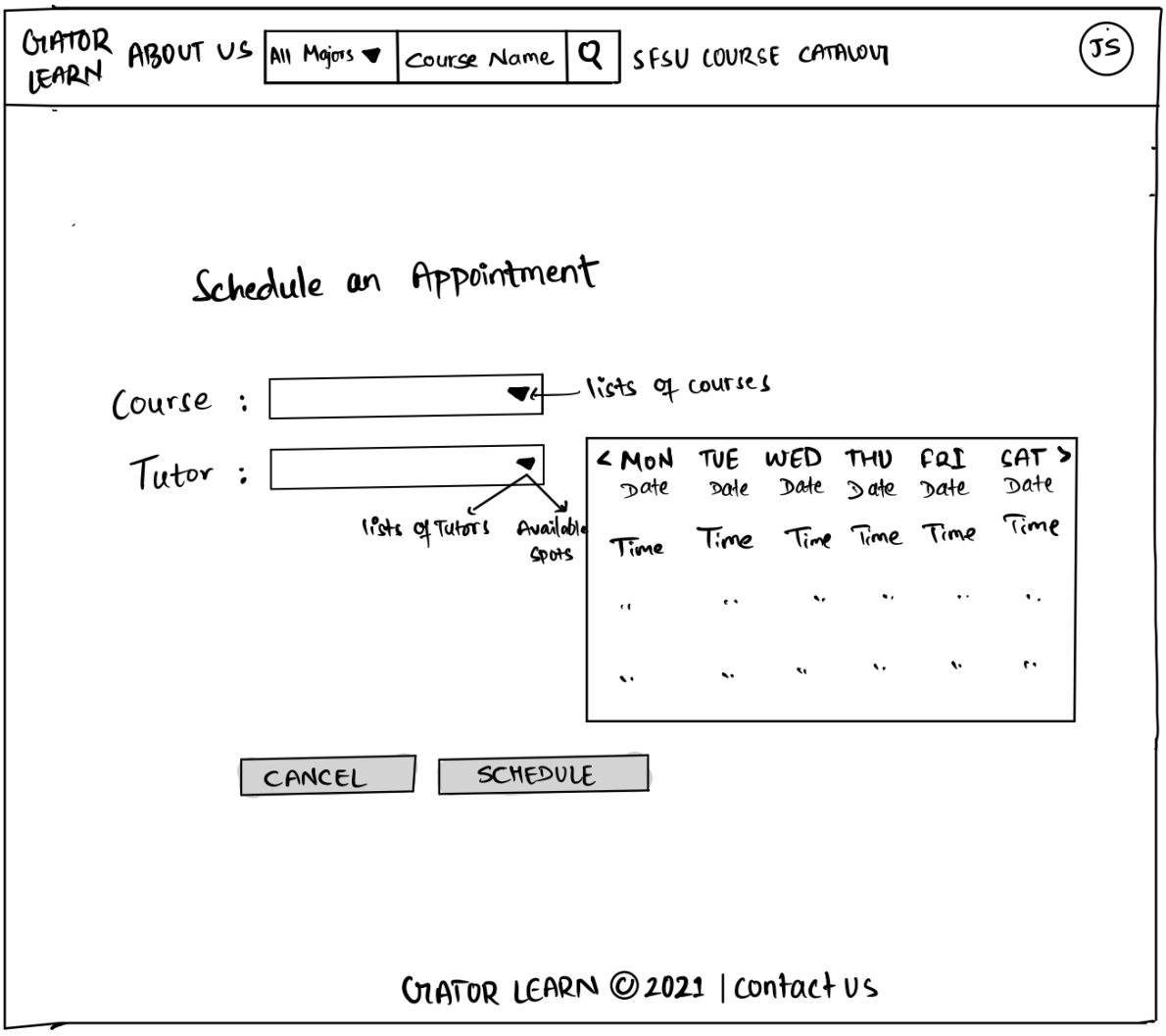
Home Page



Login Page (As already a registered user)



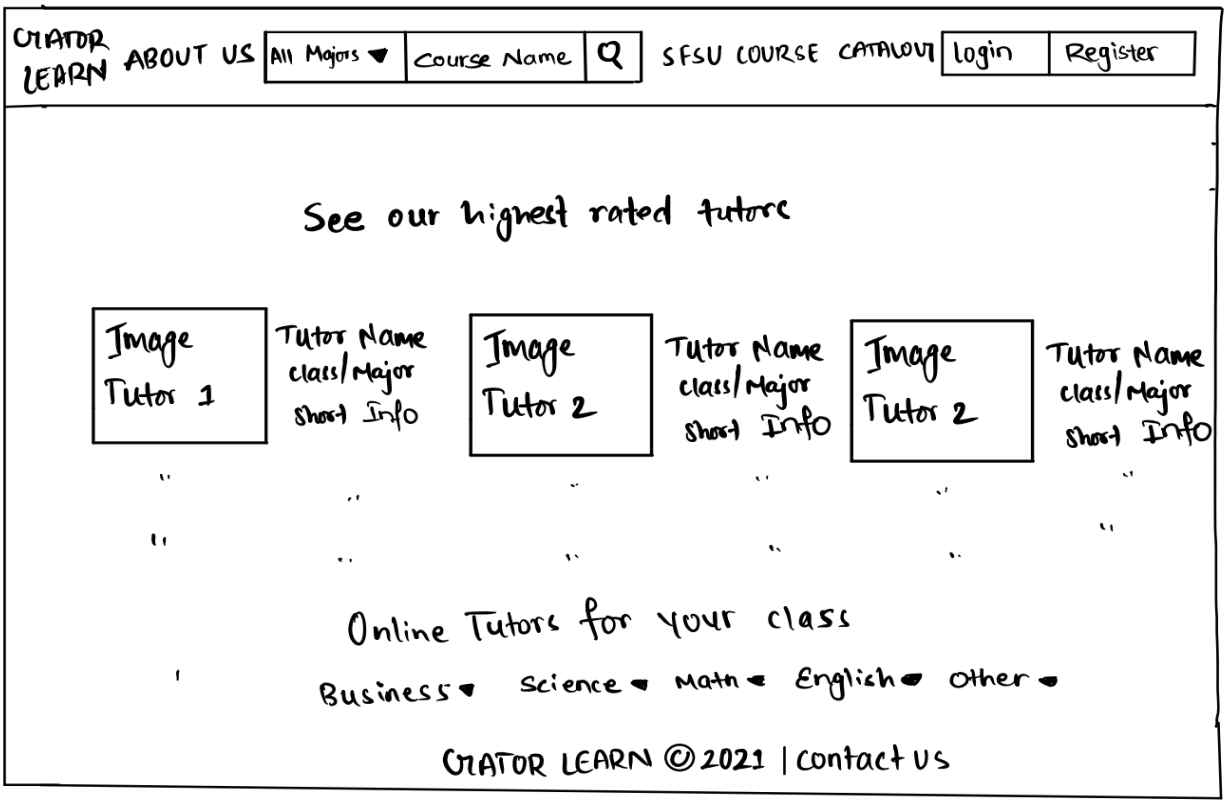
Forgot Password (Incase user doesn’t remember the password)



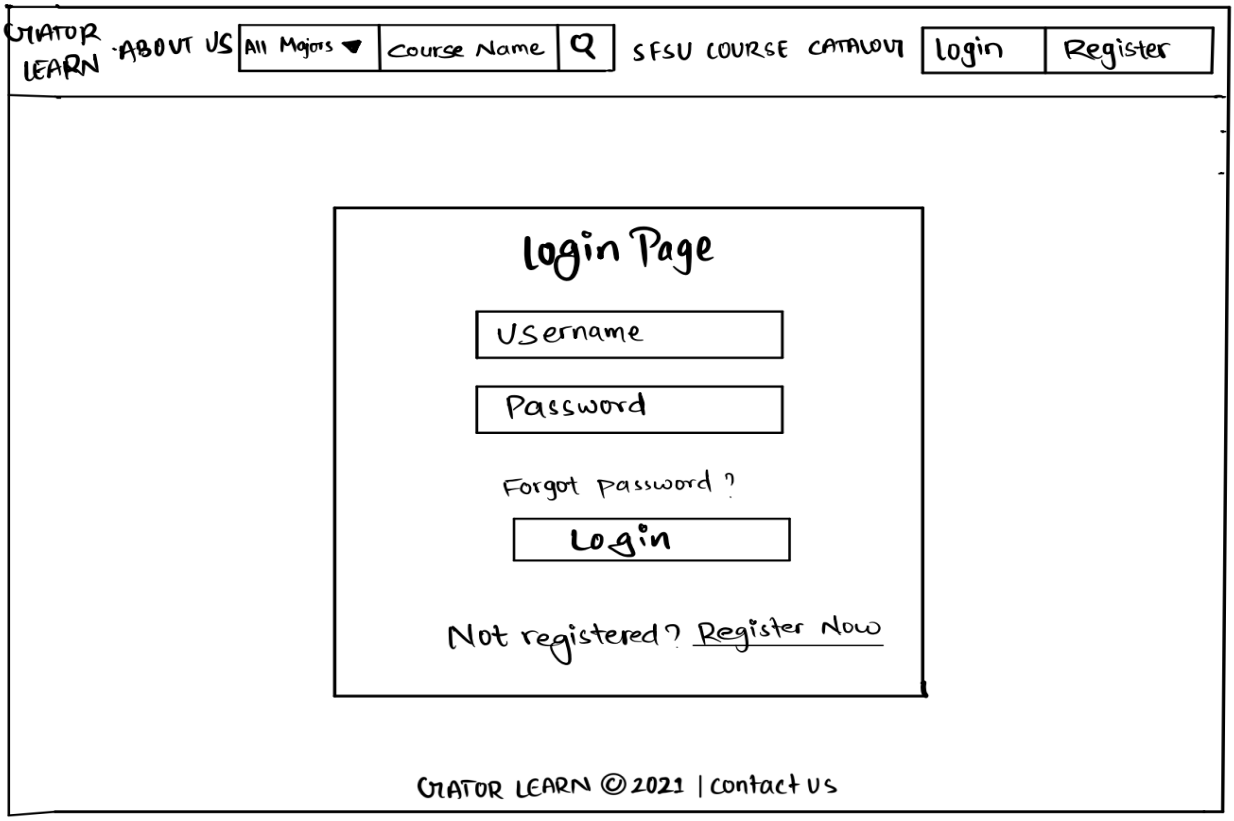
Schedule Appointment with Tutor

4. Managing users by Site Admin

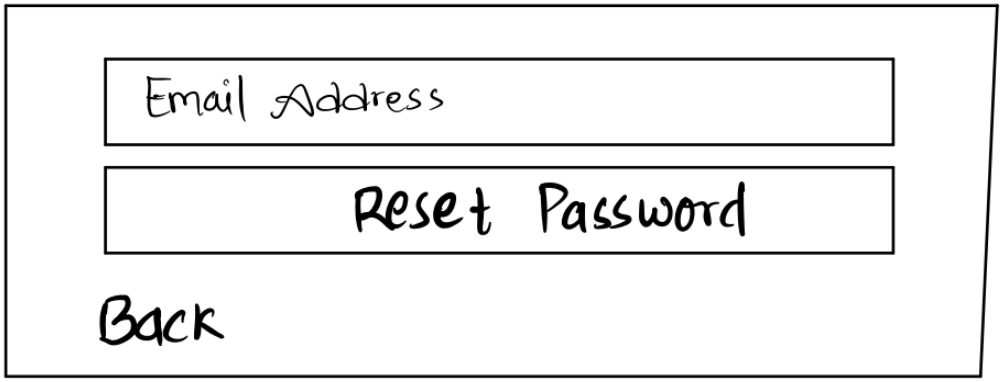
Donovan is an admin at "Gator Learn" who was also a professor at SF State University. He has an important role in this tutoring application. He has the permission to approve or decline the new users (students/tutors). He supervises whether or not the users have met all the requirements. He has access to the database of the application and manages the data and list of tutors and students on the website. However, he will not be able to edit/alter the details about users. He gets notified when the users change anything in their account and checks his inbox for the next wave of applicants to judge.



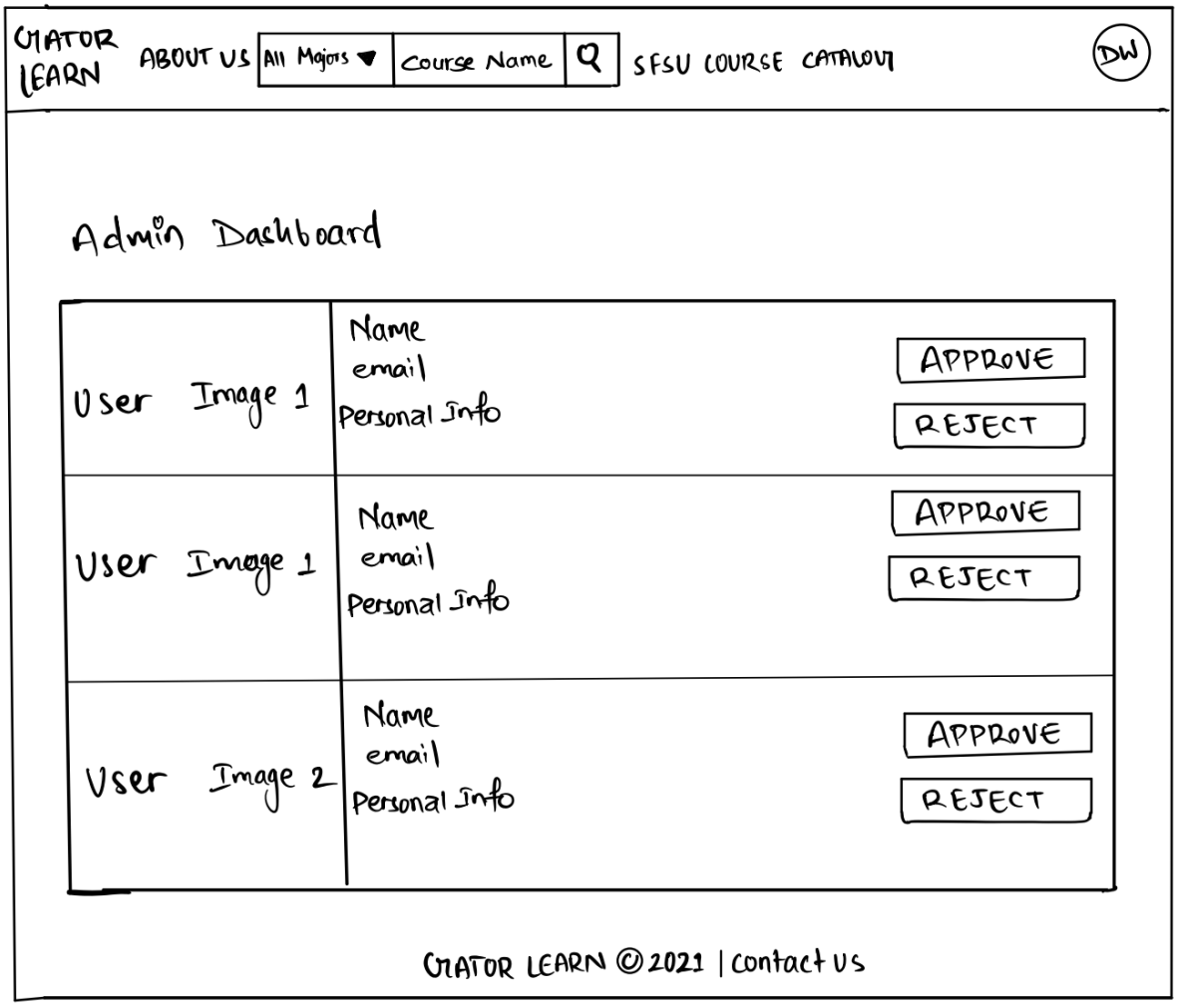
Home Page



Admin performs login



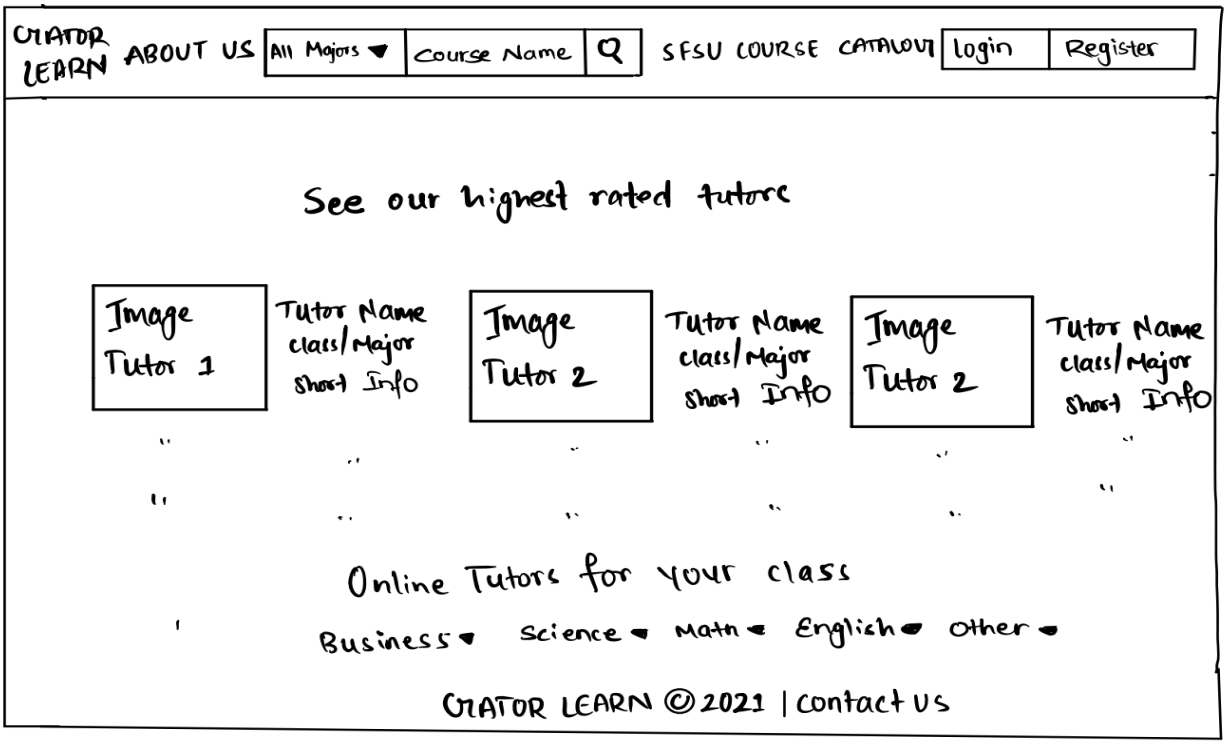
Forgot Password (Incase user doesn't remember the password)



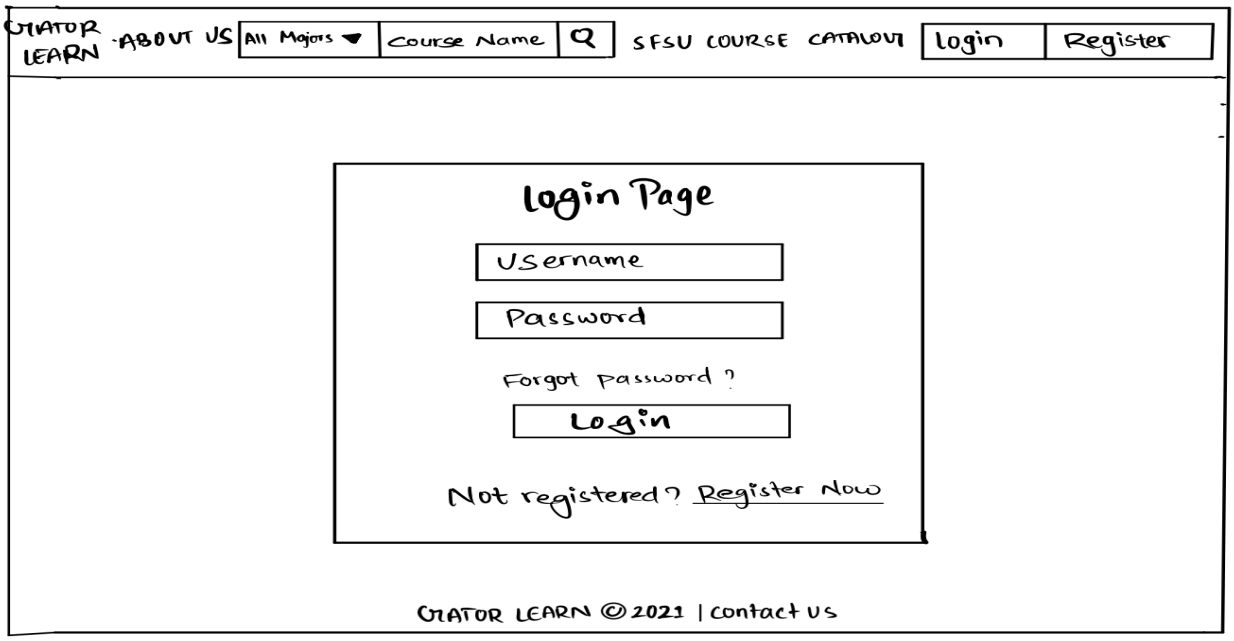
Admin Dashboard to Approve / Reject Tutor Postings

5. Adding personal information as a registered user

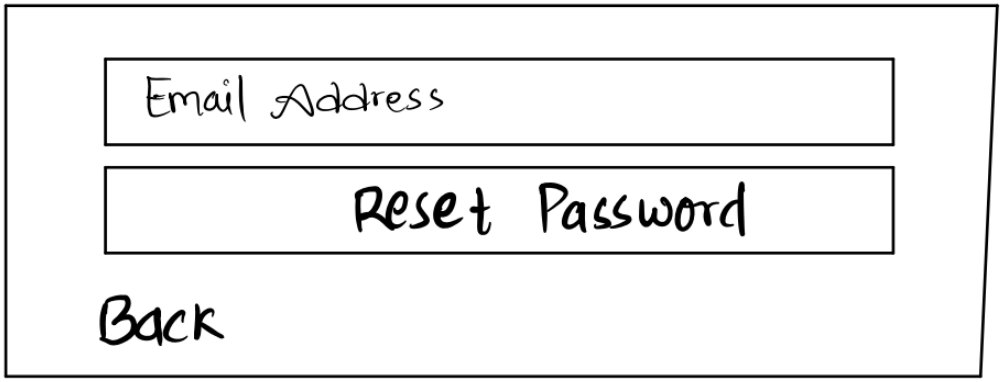
Jane, Josh, Janet and Donovan are registered and valuable users of "Gator Learn". They want to add their personal information into their profile so that everyone in the website can know about them from the profile. Once they log in and go to their dashboard, they will find a place to add their bio. They write something about them and submit their information and wait for the admin's approval.



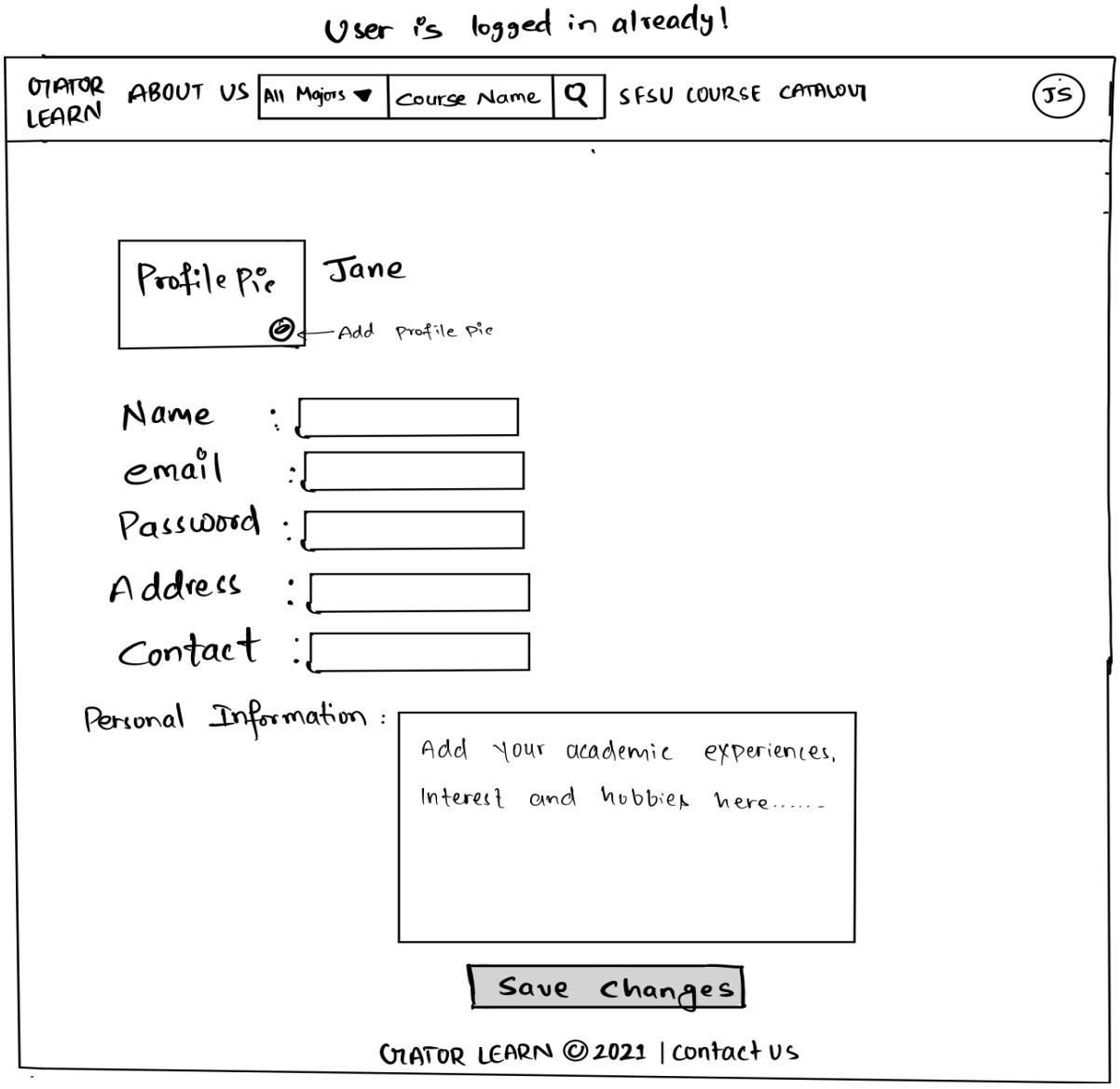
Home Page



Login Page (As already a registered user)



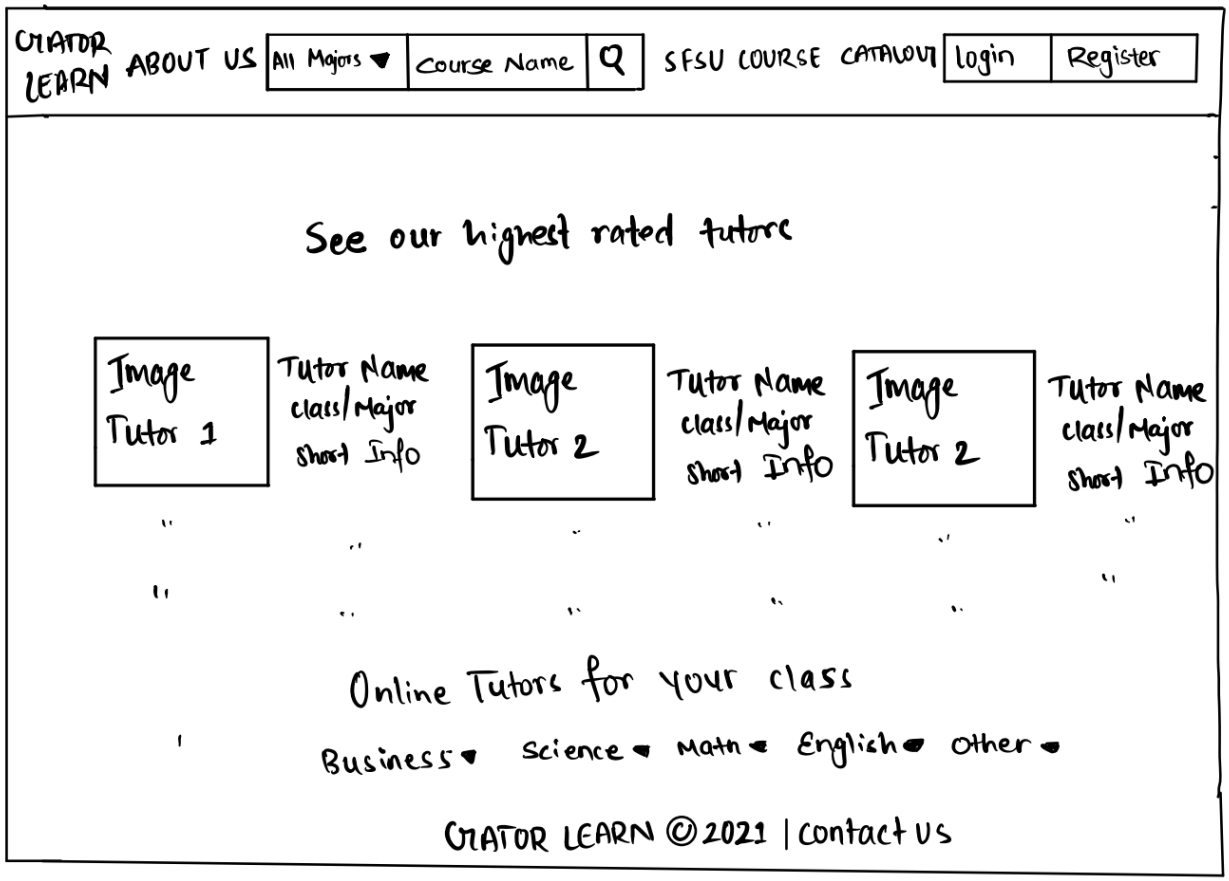
Forgot Password (Incase user doesn't remember the password)



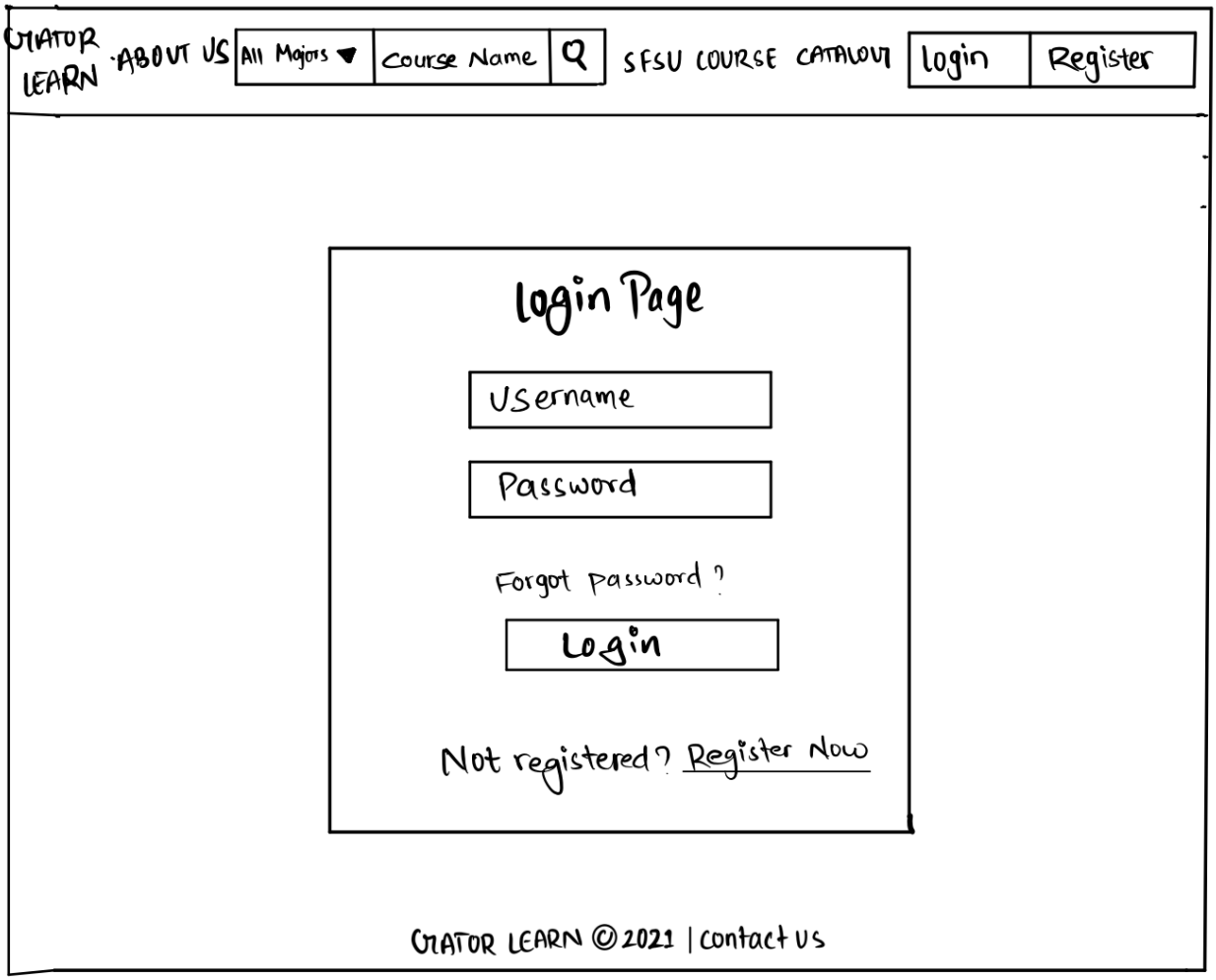
Posting (Uploading information)

6. Reviewing tutors as registered students

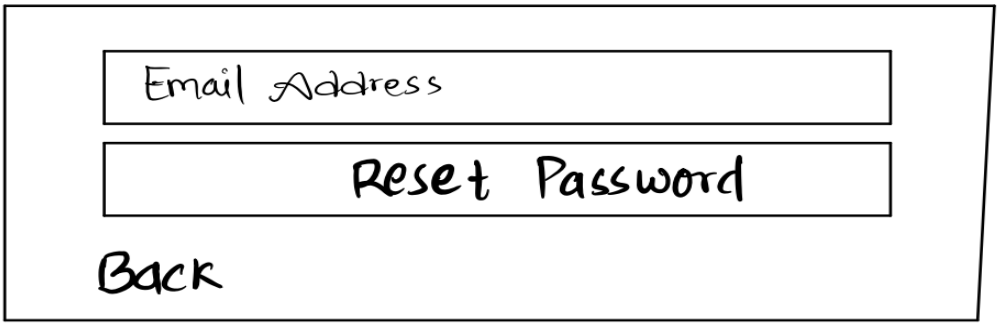
Jane and Josh were able to schedule an appointment and get the tutoring help from their respective course tutors. After the tutoring sessions, they wanted to rate their tutors because they can help other students decide who they can go to for study help. They enter their rating, and are prompted to register/sign in before proceeding. Therefore, on our website students can also leave feedback and suggestions for the tutors. If any of the tutor gets a very bad rating and review from students, the admin has a full right to delete their account from the website thereby removing the tutor.



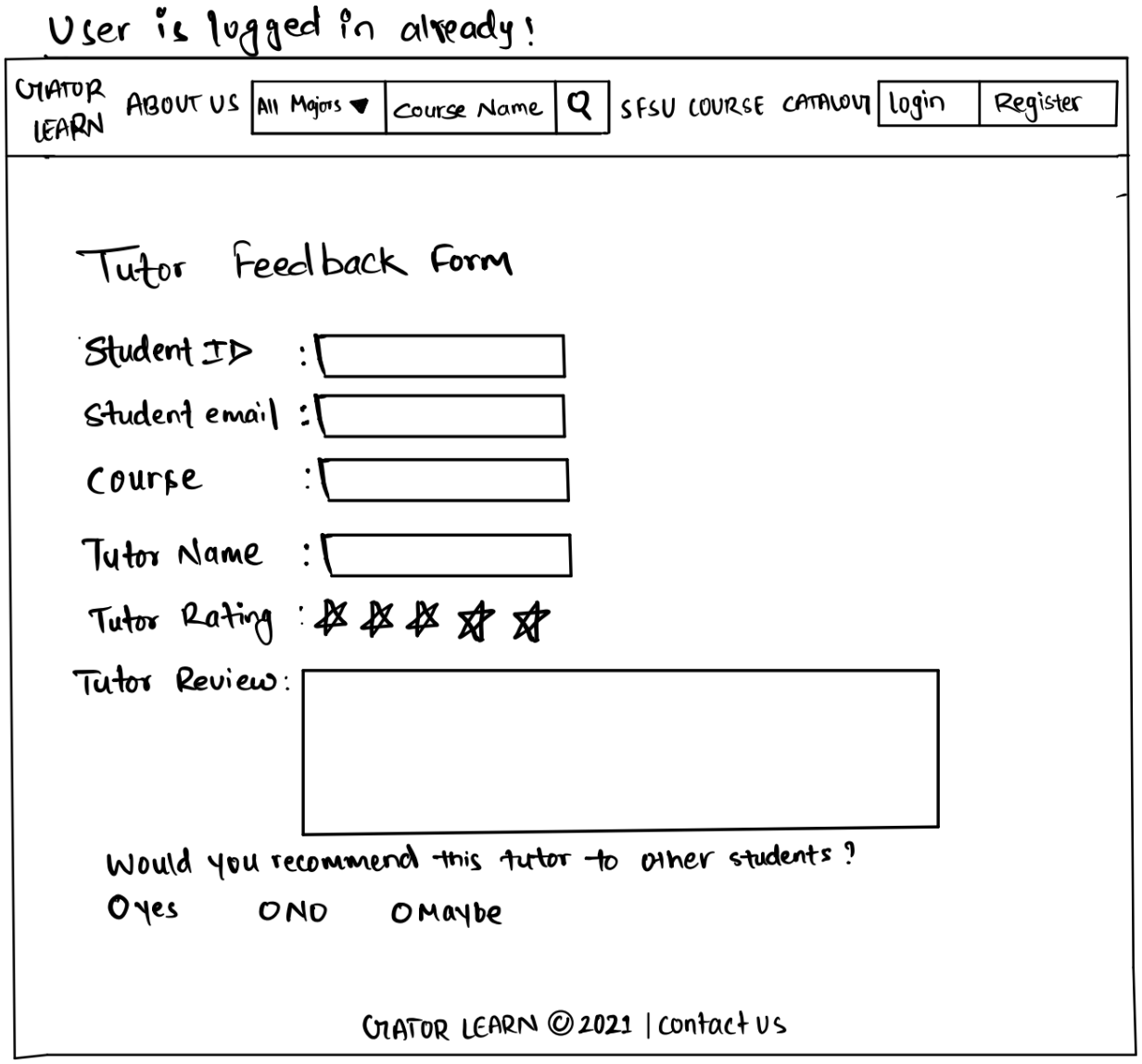
Home Page



Login Page



Forgot Password



Feedback Form

5. High Level Architecture

DB Organization

1. **Major**
   - majorId (Primary Key)
   - majorName

2. **Courses**
   - courseNo (Primary Key)
   - courseName
   - availability

3. **Catalog (Relationship Table)**
   - majorId (Primary Key and Foreign Key)
   - courseNo (Primary Key and Foreign Key)
   - requirement

4. **Unregistered User**
   - sessionId (Primary Key)

5. **Registered User**
   - userId (Primary Key)
   - sessionId (Foreign Key)
   - majorId (Foreign Key)
   - firstName
   - lastName
   - dateOfBirth
   - gender
   - email
   - password
   - avatar
   - description
   - status
   - role
   - courseTaught

6. **Tutoring_Session**
   - tsId (Primary Key)
   - tutorId (Foreign Key)
   - majorId (Foreign Key)
   - courseNo (Foreign Key)
   - status
   - date
   - timeslot
   - location
   - review

7. **Student_Session (Relationship Table)**
    - sfsuId (Primary Key and Foreign Key)
    - tsId (Primary Key and Foreign Key)

8. **Messaging**
    - msgId (Primary Key)
    - tutorId (Foreign Key)
    - sfsuId (Foreign Key)
    - sender
    - date
    - time
    - subject
    - content

Media Storage

The web application follows MVC Architecture in Python and has a static folder which will contain all the images in formats like .png, .jpg, .gif
The images are stored in a database with their relative path in Blob format with varBinary(max) datatype.

Search/filter architecture and implementation

The search bar in the web application follows this architecture: A dropdown showing the list of all the majors available in SFSU university. A text field where course name can be typed and we can get a list of all the matching courses available for that particular major or generally.

If we do not type in any course name in the text field or select any major and just click on the search button, then all majors with their respective courses will be shown to the user. If the user does specify the major and course name, a results page will be displayed showing X out of Y results based on the search (with the ability to toggle the number of items shown per page)

While typing in the course name in the text field, when you type some characters, related courses will be displayed in the dropdown by using the "% LIKE" concept in the query while coding. This LIKE clause will allow us to select specific values to return from the DB based on what the user is searching. The text that is searched will remain persistent in the text field after the search is complete and returned and results will be displayed accordingly.

APIs

1. loadHomePage(): This function will load the first page of the website i.e the homepage on clicking the url at runtime. Hereafter, you can start browsing the website.

2. bookAppointment(): This function will be used by students to book appointments with tutors according to the courses they choose. They can schedule the timings according to the tutor's availability.

Processes Description

There will be a few non trivial algorithms/ processes: A rating system for students, a feedback system for students and tutors, and ordering of the courses when displayed on any relevant page such as the search page or user home page. The rating system will be on a 5 point scale where students can rate their professors between 1 to 5 based on multiple criteria. The feedback system for registered users in general will allow users to provide feedback on the overall application system, personal preferences and other suggestions spanning a variety of potential issues. Courses will be organized and presented based on the nature of the page (search, home etc.)

6. Actual Key Risks

1. **Skills risks** (do you have the right skills)
    - Team needs to learn and get familiarize working with Flask framework and follow MVC Architecture pattern
    - Team needs to follow proper coding style
    - Unfamiliarity with AWS

    - **Plan to Resolve**
    - Team is learning and following the resources and tutorials of Flask and MVC
    - Team is following a document based on single coding style throughout the project

2. **Schedule risks** (can you make it given what you committed and the resources)
    - Team might find it difficult sometimes to do their assigned work on daily basis given unexpected events and other things on the plate.

    Plan to Resolve
    - Trying to keep small scope for the start so that the team complete their assigned work upto the deadline.

3. **Technical risks** (any technical unknowns to solve)
    - Remote hosting might be an issue sometimes with Putty as sometimes the session goes inactive and the site goes down.
    - Github organization

    Plan to Resolve
    - Its resolved for now using some commands to keep the session active all the time.
    - Learning Git and following documents related to it to create and maintain different branch

4. **Teamwork risks** (any issues related to teamwork)
    - Internal Communication gap sometimes.
    - Frontend and Backend team pacing at their assigned work so that neither of them has to wait if the work is interlinked between the two

    Plan to Resolve
    - Team coordinates within themselves to help each other and keep updated by having internal zoom meetings eg. within front-end team members and back-end team members individually, also by updating all on team discord group and Trello

5. **Legal/content risks** (can you obtain content/SW you need legally with proper licensing, copyright).
    N/A

7. Project Management

Team coordinates with each other using Discord, Trello and Zoom meetings. The agenda, the work assigned to individual team members and the deadlines is all discussed in the meeting and posted altogether on Discord and Trello to keep everyone updated about the same. There are several channels created on the Discord server group based on tasks which makes easy access for everyone in the team to look up to and complete their assigned tasks. Once the team member is done with the individual assigned work, he/she updates it on trello to let everyone else know and then also help others if something is left to do before the deadline.

Except for the mandatory zoom meeting after the class, we have our own individual team meeting to discuss the doubts and work further along the project. Also, if the work is interlinked between team members, they schedule internal team meetings on their own to discuss the same.

When the assigned task is completed team members make sure to upload it to the Google Drive folder created to coordinate between the team. Once finalized, it is then committed on Github. Team members make sure to complete the task assigned and upload the final edited version of it at max a day or two before the deadline so that the document master can just merge all of it, properly format it and submit it on/before time.