SW Engineering CSC648/848 Fall 2025

Project Title: *LEMN SFSU — Learn Easily, Mentor Naturally*

Team 06

Role(s)	***Members***	***Email***
Team Lead	*Mukisa Lubega*	*922859259@sfsu.edu*
Technical Officer	*Grishma Thumar*	*gthumar@sfsu.edu*
Github Master	*Anurag Sanadi*	*asanadi@sfsu.edu*
Back-End	*Tony Haung*	*zhuang23@mail.sfsu.edu*
Front-End Lead	*Abinash Shrestha*	*ashrestha@sfsu.edu*

Milestone 2 History

Version	***Date***	***Notes***
Submission Date	*10/31/2025*	*Submitted for m2 p1 grading*
Revision Date	*11/05/2025*	*Revised for m2 grading*

Prepared by Team 06 for CSC 648-848 Fall 2025 — San Francisco State University

Executive Summary

Access to quality academic support can make the difference between students struggling or thriving in their courses. At San Francisco State University (SFSU), many students seek tutoring but face barriers such as unreliable external sites, high costs, or scattered communication tools. **LEMN SFSU** is designed to address this gap by offering a centralized, secure, and student-exclusive platform for peer tutoring. By creating a trusted campus network, this platform exclusively for SFSU supports the schools mission to promote academic success, equity, and community building.

LEMN SFSU provides a seamless way for students to **browse and search for SFSU specific tutors** by subject, class, or availability, ensuring they find the right peer support quickly. Each tutor and student has a **verified SFSU profile**, including course expertise and short introductions, creating a reliable academic environment. A built-in **messaging system** enables secure communication without relying on external email. Students can **schedule meetings** easily and manage their tutoring activity through a **personalized dashboard** that includes scheduling tools, video postings, and messaging.
For oversight and quality assurance, **administrators manage user approvals, monitor content, and maintain community standards**, ensuring the platform remains safe and professional. An **About Page** transparently shares the project's purpose and team, reinforcing trust and clarity.

Unlike generic public tutoring platforms, LEMN SFSU is **exclusively for SFSU students**, requiring a verified @sfsu.edu email for registration. This builds a trusted peer-to-peer learning space aligned with SFSU's academic culture. The platform integrates **SFSU-specific course filters and class codes**, making it easy for students to find help tailored to their exact classes. LEMN SFSU is **completely free**, with no ads or payment systems, ensuring that academic support is accessible to all students. Administrative oversight ensures compliance with CSU IT and privacy policies, protecting both students and tutors.

LEMN SFSU delivers **academic value** by improving student performance and retention through accessible, peer-driven tutoring. It provides **social value** by strengthening the campus community, encouraging inclusivity, and fostering connections between students. It also offers **career value** by allowing tutors to develop teaching and leadership skills, showcase their experience on résumés, and open doors to future opportunities in education and mentoring.

LEMN SFSU is developed by a team of passionate SFSU software engineering students committed to solving real campus challenges. This project represents not only a technical solution but also a student-driven initiative to build a stronger academic community. With your support, we aim to make LEMN SFSU a lasting resource that enhances student success across SFSU.

List of Main Data Items & Entities

List of entities:

- Video: Tutoring videos posted by registered users
- Subject: Branch of knowledge that is being tutored by the registered user
- Class: The specific class that a registered user can cover
- Guest User: A user who has not registered an account and does not have registered user privileges
- Registered User: A user who has registered an account and has registered user privileges
- Admin: A user who has registered user privileges and manages registered accounts
- Message: An asynchronous message sending system between registered usersHigh level Functional Requirements.

User: a person who uses the tutor application with different levels of permission

- Guest user: A user who has not register an account and does not have registered user privileges
- Registered user: A user who has registered an account and has registered user privileges
- Admin: A user who has registered user privileges and manages registered accounts

Tutor Entry: A record that is made when a registered user applies as a tutor for a specific course

- Course Name: Name of course, eg., Software Engineering
- Class Number: eg., CSC648
- Subject: The category that the course belongs in, eg., Computer Science
- Availability: The times that the tutor is available

Message Entry: A record that is made when a registered user messages another registered user.

- Sender: The user who is sending the message
- Receiver: The user who is receiving the message
- Message: Alphanumeric message sent between the users.

Post Entry: A record made when a post is made by a registered user

- Image: An image post
- Video: A video post
- Text: A text post
- Approved Post: A post that is approved can be posted

Functional Requirements - Prioritized

Priority 1

- Guest User
    - 2) Guest users shall be able to use the search functionality to find appropriate tutors based on their availability, class, and subject (SFSU unique search)
    - 4) Guest users shall be able to register for a user account (with a valid SFSU email)
    - 5) Guest users shall be able to log in to an existing user account

    - 3) Guest users shall be able to view the profiles of tutors and their details
- Registered User
    - 6) Registered users shall inherit all the functionalities available to guest users
    - 7) Registered users shall be able to log into the system
    - 8) Registered users shall be able to message other registered users
    - 9) Registered users shall be able declared their time availability, class, and subjects they can tutor
    - x1) Registered users shall be able to register as a tutor for specific subjects/courses
    - x2) Registered users shall be able to view their dashboard to check messages and postings
- Admin
    - 12) Admins shall inherit all the functionalities available to registered users
    - 16) Admins shall be able to verify registered accounts applying as a tutor
    - 18) Admins shall be able to suspend registered users
    - x3) Admins shall be able to approve posts before they go live

Priority 2

- Guest User
    - 1) Guest users shall be able to browse the tutoring website
- Registered User
    - 10) Registered users shall be able to update their profile information
    - 17) Admins shall be able to approve posts before they go live ● Admin
- Admin
    - 14) Admins shall be able to view profiles of registered users
    - 15) Admins shall be able to remove inappropriate content

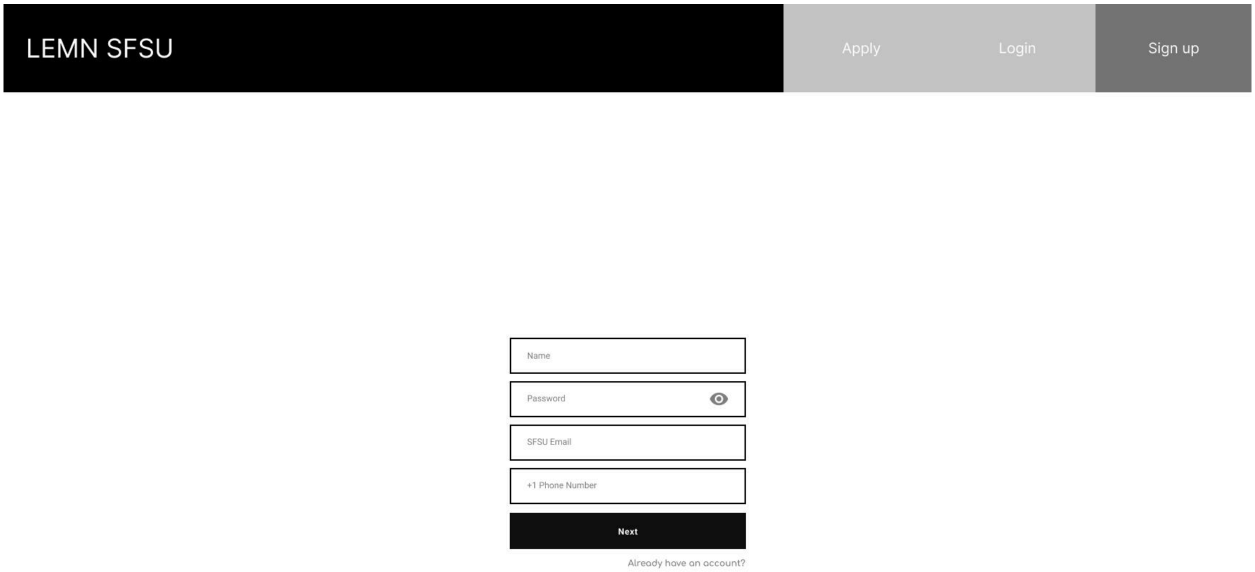
Priority 3

- Guest User
- Registered User
    - 11) Registered users shall be able to post videos
- Admin

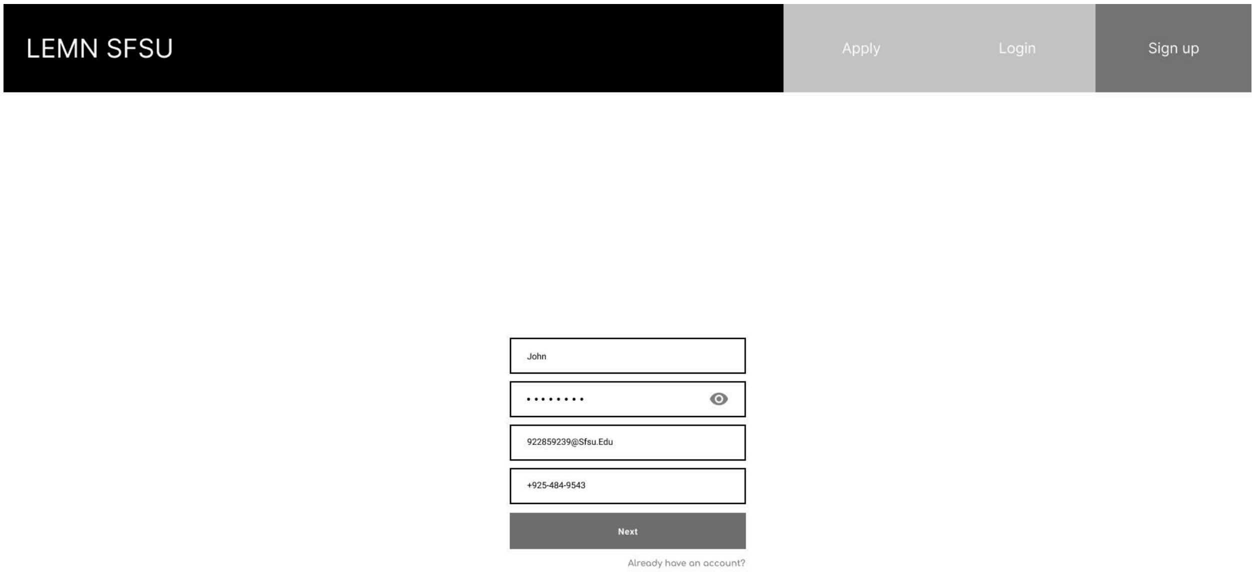
UI Storyboards

Register

Unregistered users click on Sign Up on the navigation bar. It asks for username, password, SFSU email and phone number. It also has the ‘already have an account’ for registered users.



The unregistered user enters the information username, password, SFSU email (@sfsu.edu) and phone number to register and click on the Next button to proceed.Once the information is verified the page lists the email.



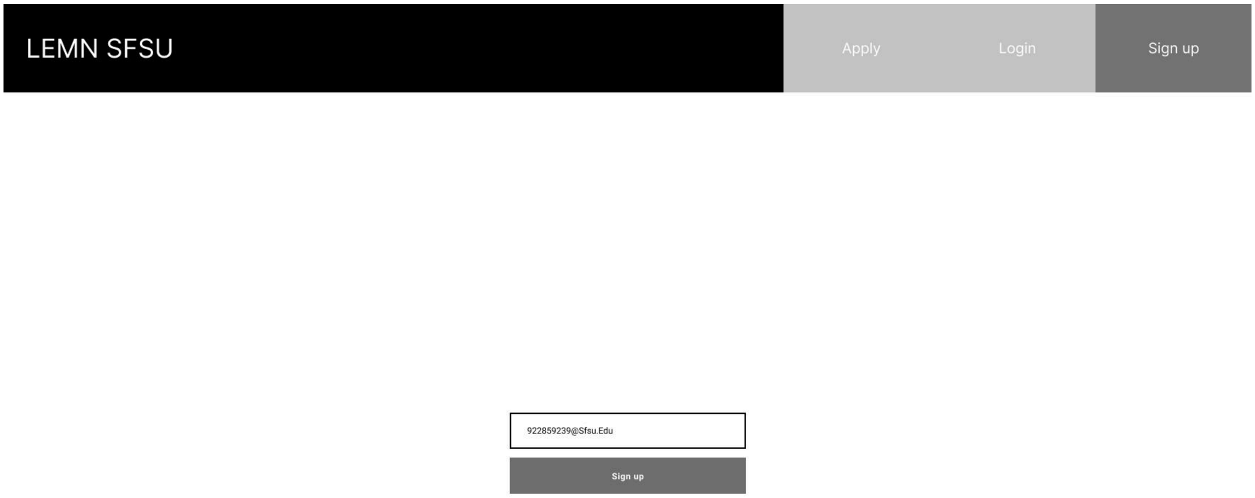
John

•••••••••

922859239@Sfsu.Edu

+925-484-9543

Unregistered users click on the sign up button to complete the registration process.



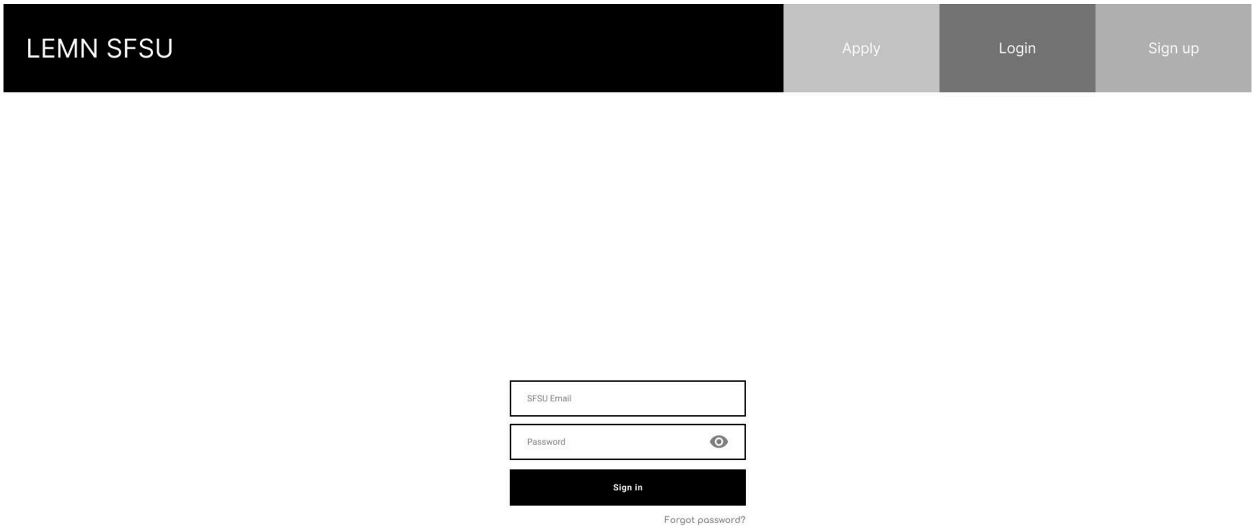
922859239@Sfsu.Edu

Login

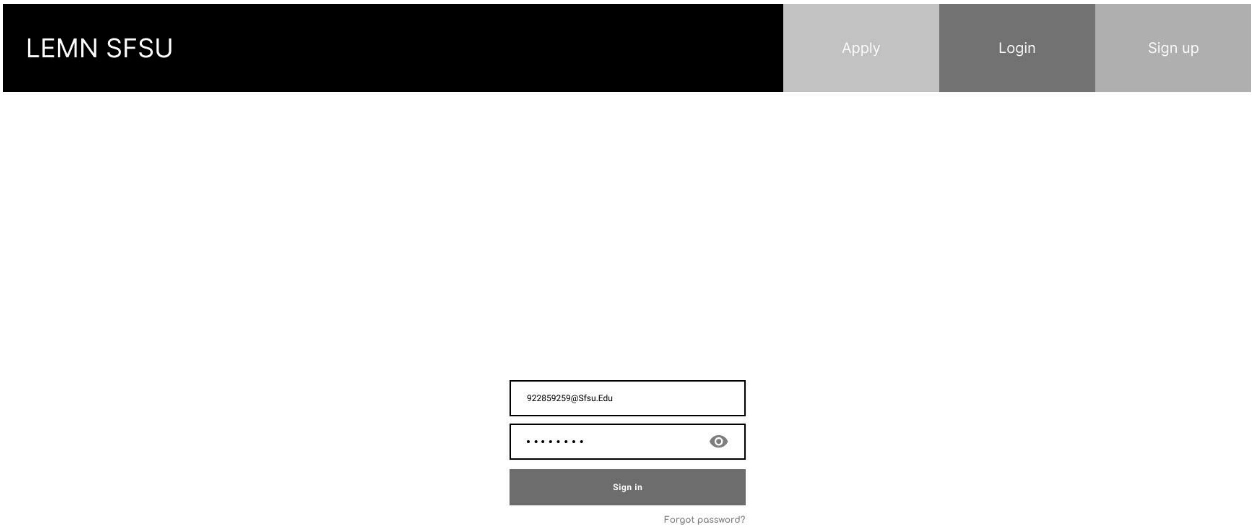
Registered users click on Login on the navigation bar.

The registered user enters the SFSU

email and password to Login / Sign in.



Once the information is entered the registered user
clicks on the Sign in button to proceed. It also has the forgot password option.The registered user enters the information to Sign in/ Login.

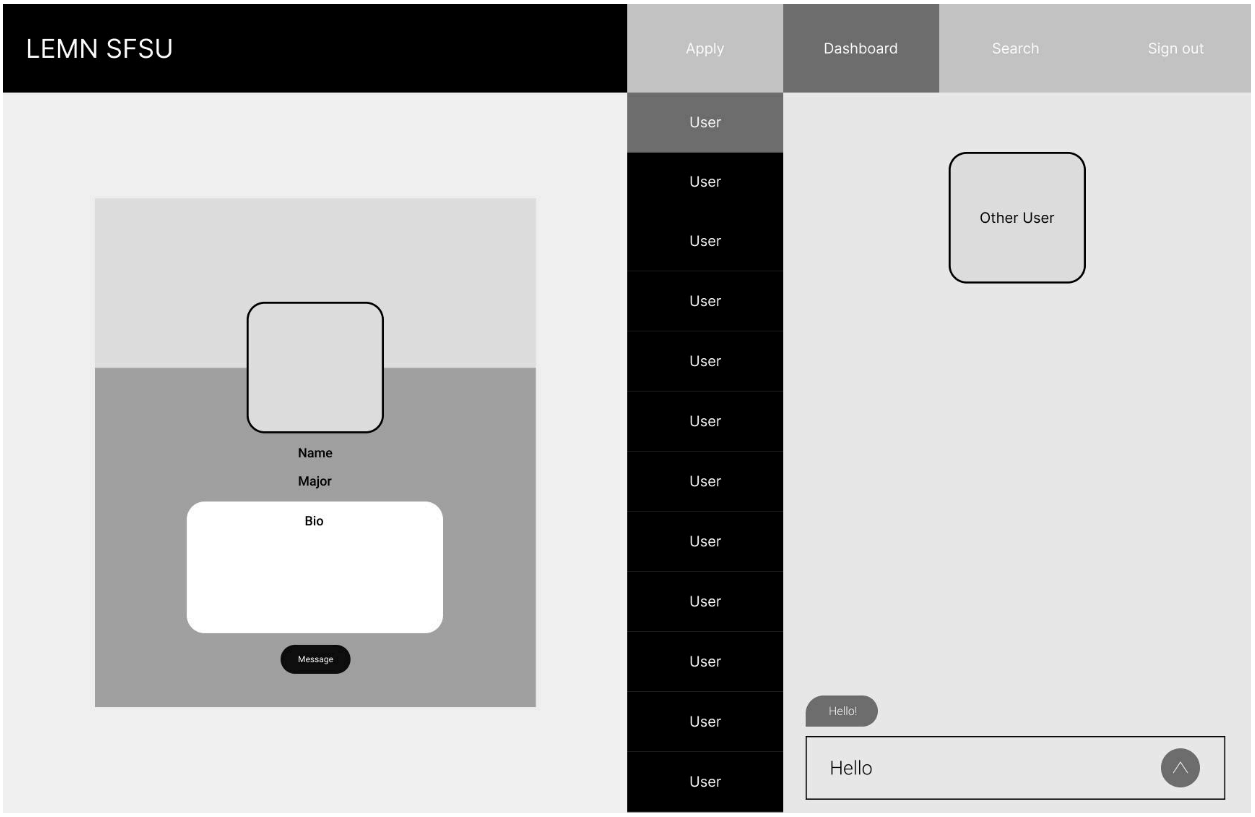


922859259@Sfsu.Edu

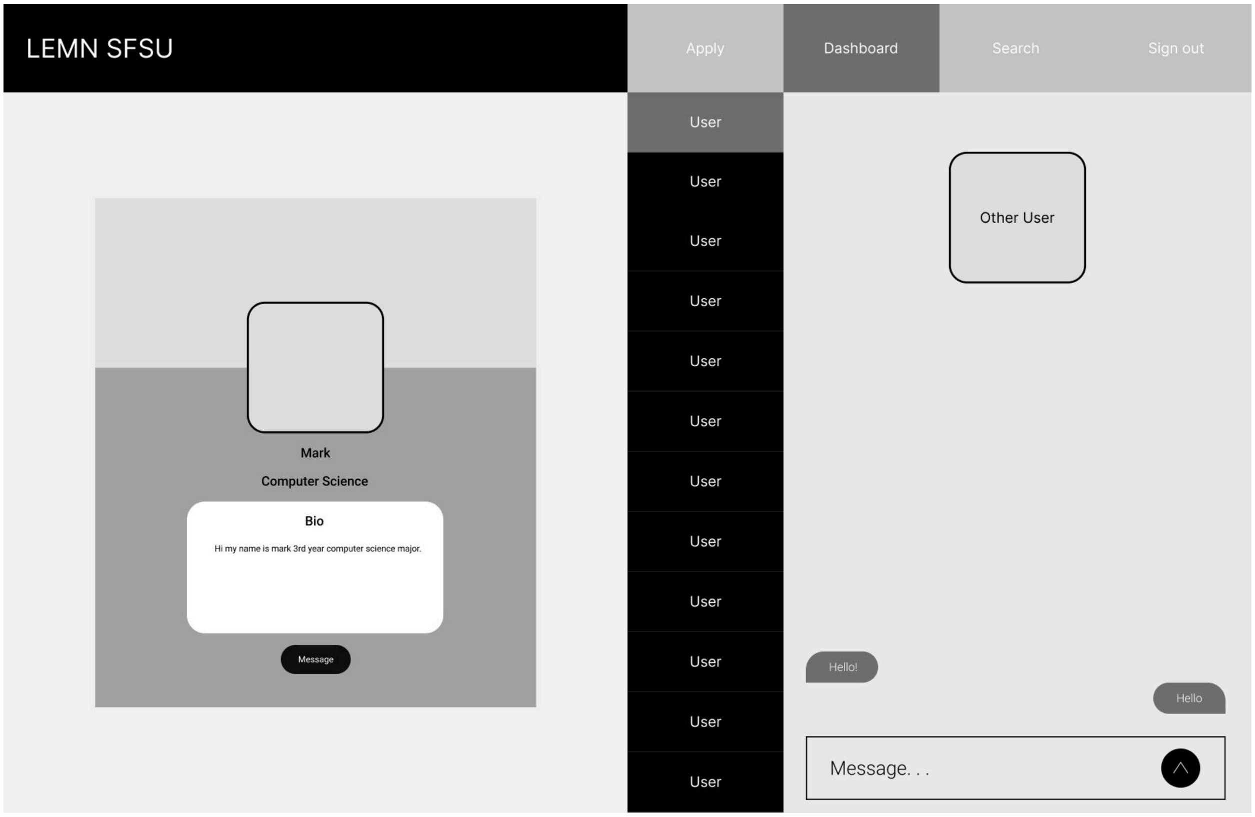
••••••••

Messaging

Once the user is logged in, it displays the dashboard. Dashboard portraits the users name, major, bio and Message button on the left hand side.



Once the user clicks on the Message
button it gives the option to select the other users to send message (depicted on right hand side).The user types in the message and clicks on send button to send the message.



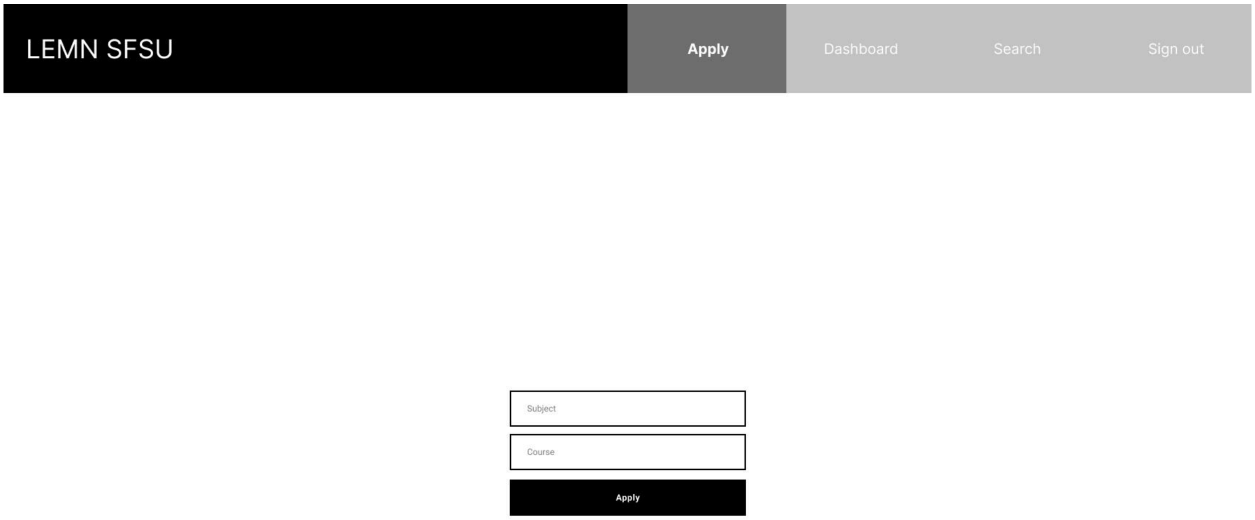
Hi my name is mark 3rd year computer science major.

Hello!

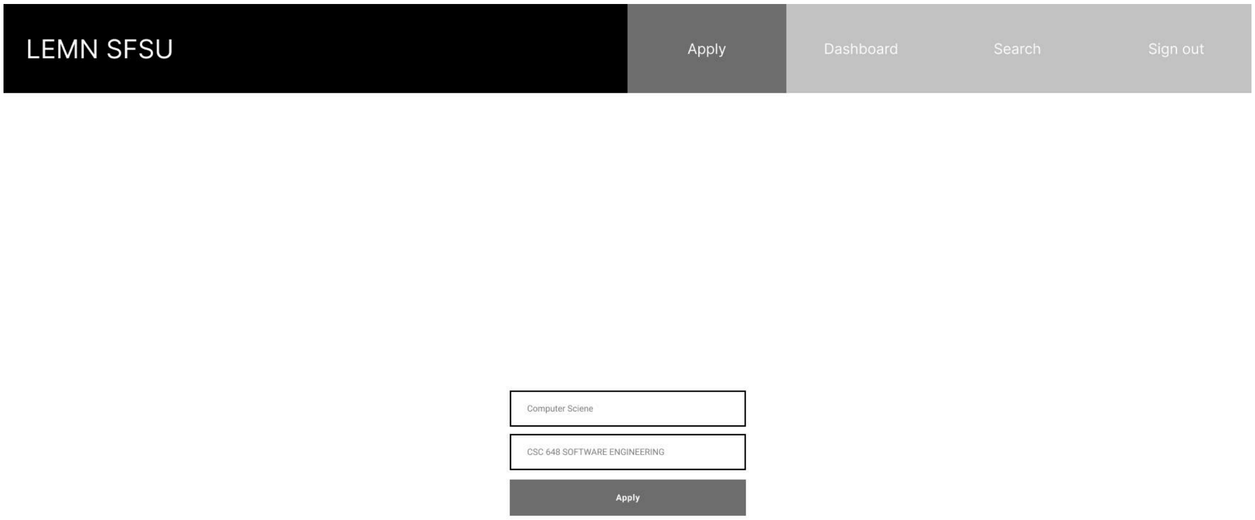
Hello

Tutor Application Form

The registered user clicks on Apply on the navigation bar. It initiates the subject and course.



The user enters the subject and course. User clicks on the Apply button to submit the form.

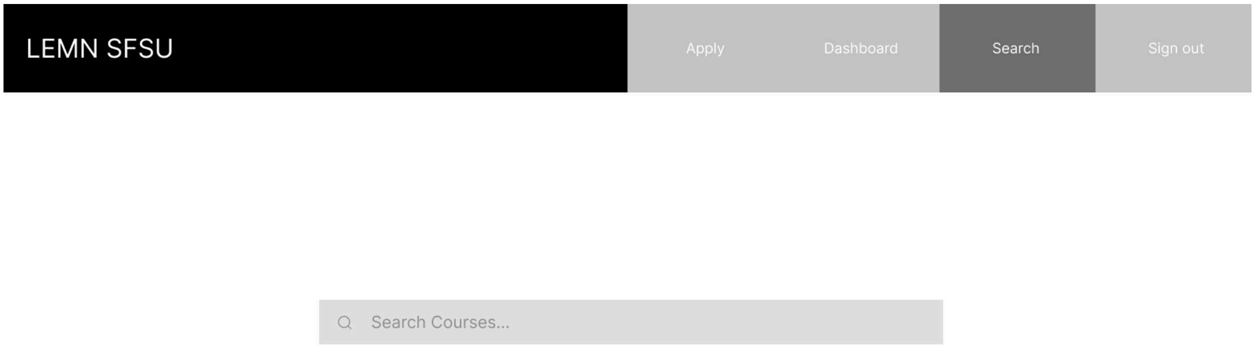


Computer Sciene

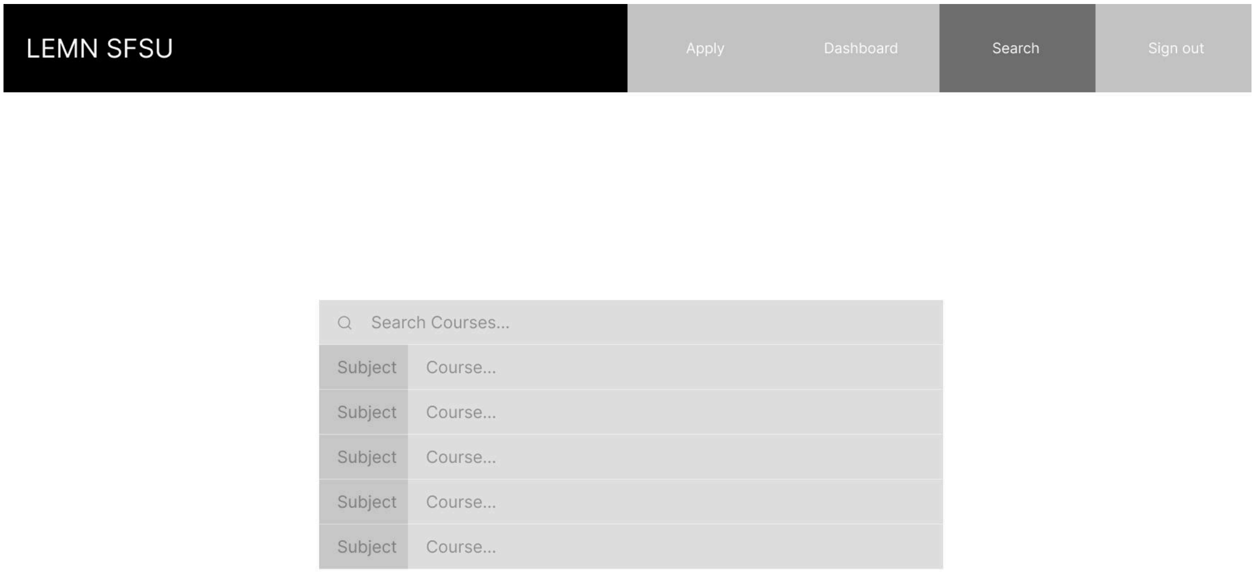
CSC 648 SOFTWARE ENGINEERING

Browsing and Search

User clicks on Search on the navigation bar. It opens up the page for browse and search (to search courses, tutors, etc.)



User types in to search. It gives a drop down list of related search results for the user to select.**2.**



High Level Architecture, Database Organization Summary

Registered User:

- Stores info about each user
- User id (PK): Numeric
- First name: Alphabetic
- Last name: Alphabetic
- Major: Alphabetic
- Minor: Alphabetic

Course: Represents each course within a subject

- Course id (PK): Numeric
- Class number (unique): Alphanumeric
- Course name (unique): Alphabetic
- Subject id (FK) -> Subject(Subject id)

User Course (junction table): links users to courses they are in

- User id (FK) -> Registered User(User id)
- Course id (FK) -> Course(Course id)
- (User id, Course id) (PK)

Subject: General academic subject area

- Subject id (PK): Numeric
- Subject name (unique): Alphabetic

Tutor Entry: Represents a tutor offering tutoring in a specific course

- Tutor entry id (PK): Numeric
- Tutor id (FK) -> Registered User(User id)
- Course id (FK) -> Course(Course id)

Message Entry: Represents a message between two users

- Message id (PK): Numeric
- Sender id (FK) -> Registered User(User id)
- Receiver id (FK) -> Registered User(User id)
- Message: Text

My suggestion is to only make the database 1NF compliant. Other normal forms are not required.

For the “Registered User” entity, the candidate key is {User_id, Course_id}

The table will look something like:

User_id	**Course_id**	**First_name**	**Last_name**	**Major**	**Minor**
1	1	Anurag	Sanadi	CS	None
1	2	Anurag	Sanadi	CS	None
2	1	Mukisa	Lubega		None
2	1	Mukisa	Lubega		None

Course_id	**Class_num**	**Course_name**
1	CSC-648	Software Engineering
2	CSC-675	Intro. To Database Systems

Key Risks

Skills Risk:

- Some team members are still developing their proficiency with React and Node.js, which may slow progress on complex features.
- *Resolution:* We plan to mitigate this by assigning tasks based on individual strengths, using tutorials and documentation for skill gaps, and pairing experienced members with those less familiar to accelerate learning.

Schedule Risk:

- Poor time management will cause delays in meetings and internal deadlines.
- *Resolution:* We will use Trello to visualize and track all tasks with clear due dates, hold weekly check-ins on Discord, and set mid-week checkpoints to ensure consistent progress.

Technical Risk:

- Integrating the front-end and back-end smoothly and managing deployment on AWS present potential technical challenges.
- *Resolution:* We will document all API endpoints early, test connections incrementally, and maintain shared debugging sessions between front-end and back-end leads.

Teamwork Risk:

- Occasional communication lags or uneven task distribution could affect workflow.
- *Resolution:* We will reinforce accountability by tracking task ownership on Trello, maintaining open communication through iMessage and Discord, and redistributing tasks when needed to balance workload.

Legal/Content Risk:

- Use of icons, images, or code snippets from external sources may pose copyright issues.
- *Resolution:* We will ensure all assets and code libraries are open-source or properly licensed and maintain documentation of all external resources used.

Project Management

For Milestone 2 and future deliverables, our team has maintained a consistent communication and coordination structure to stay organized and on schedule. We hold **weekly team meetings on Discord**, where we review progress, discuss upcoming goals, and address any blockers. Our work and documentation are stored within our **Discord server**, which serves as our central hub for file sharing, notes, and updates. For day-to-day communication and quick coordination, we use **iMessage** as our main channel for immediate responses.

As we move into Milestone 2, we plan to implement **Trello** to enhance our task management and accountability. Trello will allow us to assign specific tasks to team members, track deadlines, and visualize progress across both the front-end and back-end subteams. This will help ensure that all tasks are clearly defined, assigned, and completed on time.

To improve efficiency, we will continue to let the **front-end and back-end teams operate semi-independently** while coordinating on shared components such as API integration and database connectivity. Checkpoints and mid-week follow-ups will help us stay on track and maintain strong communication between both sides of development.

Use of GenAI

TASK	GenAI Tool	Usefulness	How It Was Used / Benefit
1. Drafting Milestone 2 Documentation	—	None	Not used.
2. Writing Project Descriptions and Objectives	ChatGPT (GPT-5)		Used to polish wording and organize the Project Management and Risk Management sections. Helped improve clarity and professional tone while keeping the team's original ideas intact.
3. Front-End Development (React / Tailwind)	—	None	No GenAI assistance used; all code written manually by the front-end team.
4. Back-End Development (Node.js / Express / MySQL)	—	None	Not used; code was developed and debugged internally by the back-end team.
5. Database / EER Design	—	None	Completed using manual diagramming and class materials only.
6. UI / UX Wireframes	—	None	Created manually by the front-end team without GenAI support.
7. Task Planning / Team Coordination	—	None	Managed through Discord and iMessage; Trello setup done manually.

Team Lead Checklist

For each item below, the team lead must answer with one of the following: DONE/OK (completed successfully), ON TRACK (in progress and expected to be completed on time), or ISSUE (problem encountered, with short explanation).

Item	Status	Comments (if ISSUE)
So far all team members are fully engaged and attending team sessions when required	OK	—
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	ON TRACK	—
Team reviewed suggested resources before drafting Milestone 2	DONE	—
Team lead checked Milestone 1 document for quality, completeness, formatting and compliance with instructions before the submission	DONE	—
Team lead ensured that all team members read the final M2 and agree/understand it before submission	DONE	—
Team shared and discussed experience with GenAI tools among themselves	DONE	—