SW Engineering CSC648-848 Fall 2025

GatorGuides

Team 09

Jason Javandel (jjavandel@sfsu.edu) – Team/Server Lead

Andy Yip – (ayip3@sfsu.edu) – Frontend Lead

Connor Moore – (cmoore14@sfsu.edu) - Database Lead

Shirish Maharjan - (smaharjan@sfsu.edu) - Github Lead

Manea Fadl Manea – (mmanea@sfsu.edu) - Backend Lead

Milestone 1

Submission Date	Revised Date
10/17	10/27

Table of Contents

Executive Summary

GatorGuides is a tutoring platform built by and for student of San Francisco State. We help simplify connecting tutors and students by eliminating the need to schedule in person, as well as providing students with a rating system to understand the tutor's teaching styles and overall feedback.

Tutors will have their own page that they can customize with a biography, the subjects they tutor, as well as a background/text color(tentative). Tutors can select days and time ranges in which they are open for appointments. These pages will be visible without any authentication, but scheduling will be restricted to users that are signed in.

Students will be able to filter subjects to be tutored in, which tutor they want, as well as a time frame, and be shown relevant results in a clean way. Students can look at reviews and their experiences with the tutor from previous students and decide who they want to select. If they do not care about which tutor or timeframe they want, they will be presented with a list of the soonest appointments to select from. After a tutoring session has concluded, the user will be given a notification asking them to leave a review on the tutor for themselves.

Our team of students is committed to providing an easy to use and hassle-free tutoring service for the students of SFSU. We aim to provide this ease-of-use without compromising looks, something that seems to be ever more common inside the world of educational software. Being students ourselves, we have unique insights and views on what other students would want out of a service like ours, giving us an advantage over existing solutions.

Personae

1. Jake (Non-Tech-Inclined Student)
    a. Uses a MacBook or Windows laptop with Safari or Chrome
    b. Accesses the app mostly from school or home Wi-Fi
    c. Basic computer literacy (web browsing, file uploads, school portals)
    d. Limited troubleshooting ability; relies on help pages or peers
    e. Wants a simple, quick sign-up process
    f. Prefers to learn by exploring rather than reading manuals
    g. Seeks an intuitive, minimal interface without extra setup
    h. Uses one password for multiple sites
    i. Spends time reading through webpages to understand features
    j. Gives up easily if the app feels confusing or slow
    k. Frustrated by complex registration steps or technical jargon
    l. Dislikes frequent password resets or multi-factor logins
    m. Finds cluttered layouts and unclear instructions overwhelming

2. Diego (Tech-Inclined Student)
    a. Use a Mac or Linux system with any browser (Chrome, Firefox, Brave, etc.)
    b. Often multitasks across multiple tabs and applications
    c. Comfortable with advanced settings, shortcuts, and troubleshooting
    d. Uses complex, secure passwords and sometimes password managers
    e. Familiar with command-line tools, coding, or system customization
    f. Wants a fast, seamless sign-up and login process
    g. Values efficiency and performance over visual design
    h. Prefers systems that “just work” without unnecessary steps

i. Skims or skips instructions; learns by doing
j. Quickly navigates through menus and options
k. Gets frustrated by slow loading or excessive confirmations
l. Annoyed by forced tutorials or long sign-up forms
m. Dislikes systems that limit customization or control
n. Loses patience with laggy, poorly optimized interfaces

3. Julia (Mobile-First Student)

a. Primarily uses her smartphone or tablet to access apps and websites
b. Occasionally switches to a laptop for larger assignments
c. Relies on campus Wi-Fi or mobile data while on the go
d. Comfortable with mobile apps and responsive interfaces
e. Knows basic device settings and app management
f. Not deeply technical but efficient at everyday phone use
g. Wants the app to be fully mobile-friendly and easy to navigate
h. Seeks quick access to key features without unnecessary clicks
i. Prefers auto-login or biometric sign-in for convenience
j. Uses her phone throughout the day for schoolwork, communication, and browsing
k. Prefers simple, swipe-based navigation and clean layouts
l. Multitasks between messaging, studying, and social apps
m. Gets frustrated when the site isn't optimized for mobile screens
n. Dislikes tiny buttons, slow pages, or constant re-logins
o. Finds typing long forms or passwords on mobile keyboards tedious

High-level Use cases

Use Case 1: Browsing and Searching of Tutors.

Jake is a student who is not very tech savvy, he visits GatorGuides to seek assistance in a computer science exam he is about to take. He searches in a list of available tutors by subject and by department. Although he is not really tech-savvy, the interface enables him to easily navigate and see tutor profiles which contain subject knowledge and ratings, as well as reviews made by other students. This assists Jake to make decisions on which tutor can accommodate him and his schedule and not lose or be overwhelmed. To him, this is so as to get a reliable, SFSU-approved tutor as soon as possible.

Use Case 2: Booking a Tutoring Event.

As a mobile-first student, Julia can use GatorGuides on her phone in-between school hours to find a tutoring session. She picks a tutor according to his or her availability, the subject, and the comments of students. She does not have to worry about having to log into various systems, and as such, she easily makes a time slot that fits her schedule. Julia enjoys the ability to book her next appointment through her mobile phone, and that all future appointments are shown on her profile. She can easily have control over her academic support even when she is in motion because the platform is flexible.

Use Case 3: Making a Rating and Review.

Once Diego has finished a successful tutoring session, he is asked to write his feedback on GatorGuides. He gives a rating and a brief written comment on whether his tutor was clear, punctual and helpful in teaching. The review prepared by Diego is added to the current community of SFSU students who use peer-reviewed feedback to get efficient tutors. He appreciates the fact that the system does not prohibit honest and constructive reviews and other students such as Jake and Julia can also make use of his experience in making decisions on which tutor to engage next semester.

Use Case 4: Tutor Reviewing Feedback and Availability.

Professor Jose is a tutor on GatorGuides and is going through the statements of students about his recent sessions. He goes through the comments to see what strengths he has with his method of tutoring and what he can do to be improved in his methods. He also makes changes to his availability in future weeks so that students can get the slots with ease. This process can assist tutors such as Professor Jose to better their schedules as they go on ameliorating the learning experience to SFU students.

Use Case 5: SFSU Student Check and Tutor Authentication (SFSU-Exclusive Option)

To become a tutor on GatorGuides, a new user has to ensure that he or she is a student of SFSU first. This is achieved by performing an authentication process by which the SFSU email and student ID are verified by the admin team. After confirmation, the profile of the user is approved and given tutoring services. This special mechanism makes certain that all tutors on the site are SFSU students that make users such as Jake and Julia assured that they are dealing with reliable associates in the university, a feature that differentiates GatorGuides among other tutoring websites.

List of main data items and entities

1. Admin – User that holds control over the registered users in the system; allows for the additions, deletion, and alterations of the website
2. Visitor – Person that has not made an account or not logged in; allows for browsing of tutors, but not allowed to schedule an appointment
3. User – Person that has made an account and is logged into; allows for browsing of tutors, scheduling, and calendar functions
4. Tutor – User approved for mentorship; allows for creation of the tutor profile, appointments, and hourly fees
5. Tutor authentication – Verification for tutors; Confirmation of SFSU attendance—email and ID—and peer review of teaching ability
6. Session – Allotted time determined by tutor for study
7. Glossary – List of all tutors, subjects, and topics available
8. Calendar – Showcases all available tutoring sessions on a weekly or monthly format
9. Tags – List of subjects or topics taught by the tutor
10. Browsing – Allows for searching of glossary
11. Scheduling – Registered user’s reserving a tutoring session

12. Boolean search – Allows for AND, OR, and NOT in filtered searches
13. Fees – Priced determined by tutors for teaching services
14. Coursing tracking – Provides the user the ability input courses under their profile, allowing for better filtering when searching for tutors

List high level functional requirements

Visitor

1. Must allow visitor to access glossary of available tutors

2. Visitor must be able to search glossary for specific subjects or tutors

3. Must allow users to create accounts

User

4. User must only be allowed to make one account

5. User cannot access calendar while signed out

6. Users must be logged in to schedule a tutoring session

7. One user must be able to make multiple sessions for different users

8. Calendar must accurately reflect any sessions scheduled until past the date

9. User must be able to cancel session and have it reflected to tutor

10. User must have the ability to create a tutor account

11. Users must verify their SFSU status through manual review to become Admin

Tutor

12. Tutor accounts must be able to make listings for their services

13. Tutor must be able to edit their page at any time (add or remove info)

14. Tutor must be able to populate their page with tags and a bio

15. Tutor must be able to see sessions scheduled with them

16. Tutor must be able to cancel session and have it reflected to user

17. Tutors must not be allowed to message users before a session is scheduled

Admin

18. Admin must be able to terminate user and tutor accounts

19. Admin must be able to delete posts

20. Admin must be able to communicate with both users and tutors

List of non-functional requirements

1. Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0
2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers
3. All or selected application functions shall be rendered well on mobile devices (no native app to be developed)
4. Posting of tutor information and messaging to tutors shall be limited only to SFSU students
5. Critical data shall be stored in the database on the team's deployment server.
6. No more than 50 concurrent users shall be accessing the application at any time
7. Privacy of users shall be protected
8. The language used shall be English (no localization needed)
9. Application shall be very easy to use and intuitive
10. Application shall follow established architecture patterns
11. Application code and its repository shall be easy to inspect and maintain
12. Google analytics shall be used
13. No e-mail clients shall be allowed. Interested users (clients) can only message service providers via in-site messaging. One round of messaging (from client to service provider) is enough for this application. No chat functions shall be developed or integrated
14. Pay functionality (e.g. paying for goods and services) shall not be implemented nor simulated in UI.
15. Site security: basic best practices shall be applied (as covered in the class) for main data items
16. Media formats shall be standard as used in the market today
17. Modern SE processes and tools shall be used as specified in the class, including collaborative and continuous SW development and GenAI tools
18. The application UI (WWW and mobile) shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Fall 2025. For

Demonstration Only" at the top of the WWW page Nav bar. (Important so as to not confuse this with a real application).

Competitive analysis

Features	Tutor.com	wyzant.com	SFSU TASC	GatorGuides
Tutor authentication	+	+	+	++
Browsing	+	+	-	++
Scheduling	+	+	+	++
Boolean Search	-	-	-	+
Course Tracking	-	-	-	+

Keys: Exists: + Exceeds: ++ Void: -

Summary:

One of the advantages that we hope to bring in our application is the tutor authentication, ensuring that aspiring tutors are peer-reviewed and respectful of their mentee. Another notable feature is the ability to have a personalized schedule for courses that you are currently enrolled in to match the needs of registered users, streamlining the process to find a tutor. Browsing, scheduling, and searching are made to be easy to navigate, but also specific enough for registered users to find what they want.

High-level system architecture and technologies

SW components and versions

Database

- MySQL 9.4.0

Deployment cloud servicer planned to use

- AWS 1 CPU 1 GB RAM

Front end frameworks planned to use

Frontend

- SvelteKit 2.22.0 (routing)
- Svelte 5.0.0

- Tailwind CSS 4.0.0

Backend

- Python 3.13.7
- FastAPI 0.119.0

Browsers planned to support
Google chrome (139-141) and Mozilla Firefox (142- 144)

Major additional external open-source APIs planned to use

Use of GenAI Tools

GenAI was used to come up with a name for our application, as well as giving an outline for defining the personae's.

Team Checklist

- So far all team members are fully engaged and attending team sessions when required
    - DONE/OK
- Team found a time slot to meet outside of the class
    - DONE/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
    - DONE/OK
- Team reviewed class slides on requirements and use cases before drafting Milestone 1
    - DONE/OK
- Team reviewed non-functional requirements from “How to start…” document and developed Milestone 1 consistently
    - DONE/OK
- Team lead checked Milestone 1 document for quality, completeness, formatting and compliance with instructions before the submission
    - DONE/OK
- Team lead ensured that all team members read the final M1 and agree/understand it before submission

    - DONE/OK
- Team shared and discussed experience with GenAI tools among themselves
    - DONE/OK
- Github organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.)
    - DONE/OK