SW Engineering CSC 648-848 Fall 2024

StudyGator

Team 08

Team Lead: Bryan Lee (blee37@sfsu.edu)

Front-end lead: Min Ye Thway Khaing

Back-end lead: Nishi Suratia

GitHub master: Kenneth Wen

Milestone 1

Date Submitted	Date Revised
10/11/2024	10/13/2024

1. Executive Summary

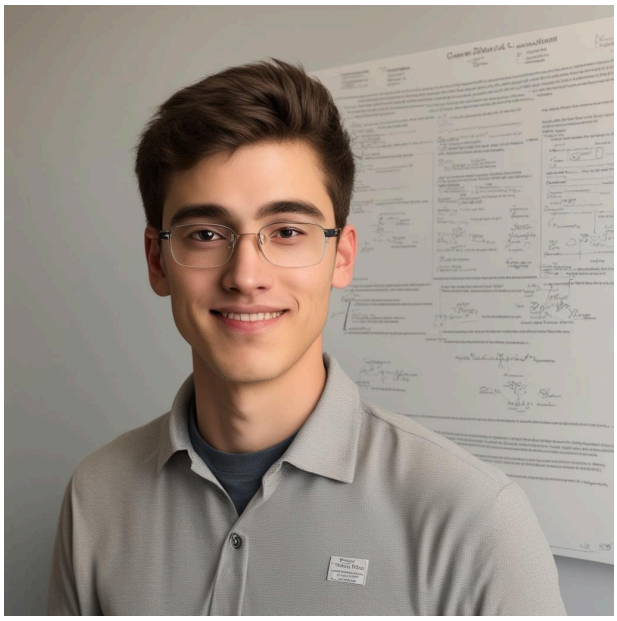
As the university semester progresses, it becomes increasingly demanding and hectic. Balancing the many coursework, project deadlines, and exam preparation can become overwhelming and stressful, especially with the limited time students typically have. Traditional options to seek academic assistance, like office hours, are often inadequate as they could conflict with other classes, extracurricular activities, or part-time jobs. In this rapid academic environment, students need flexible, easily accessible resources tailored not only to the specific course content but also to the unique teaching styles and expectations of each professor.

This is where our application, StudyGator, actively addresses the challenge of finding academic support tailored to specific courses or professors at SFSU. Users no longer have to spend hours looking through materials they don't understand, as StudyGator connects them directly with experienced SFSU tutors or alumni who are familiar with their specific coursework. This personalized approach not only streamlines efficient studying but also alleviates the academic burden, allowing students to tackle their coursework with confidence. StudyGator is built explicitly by and for SFSU students, offering the unique feature of allowing users to browse tutors offering services for specifically listed SFSU subjects and courses. StudyGator ensures that help is always just a click away, allowing SFSU students to focus on what matters most–their academic success.

Our team is a diverse group of students seeking to connect SFSU students with tutors more efficiently. Equipped with extensive backgrounds in software development and user-oriented design, we plan to deliver a tutoring application focused on simplicity and ease of use. Our focus is on creating a seamless tutor browsing experience and a way for SFSU tutors to put themselves out there. We believe that finding a tutor shouldn't feel like shopping for a new outfit—it should be quick, straightforward, and flexible enough to fit into students' hectic schedules.

2. Personae

John Johnson (User)



John Johnson is an SFSU student who is currently struggling in his organic chemistry class. He is proficient with computers and plays video games with his friends in his free time. He has a part-time job, so he has some money to afford tutoring; however, his budget is tight.

Goals:

- Passing his organic chemistry class
- Easily be able to access help from knowledgeable people

Pain Points:

- Budget-conscious
- Does not have time to sift through reviews and browse endlessly for tutors
- Stressed with little free time

Mary Merriam (Admin)

Mary Merriam has been a computer science professor at SFSU for over ten years. She is happy to help students and is willing to spend her free time helping them improve. Professor Merriam enjoys reading books and spending time with her two dogs. She is also familiar with MySQL workbench and has taught it in her lectures.

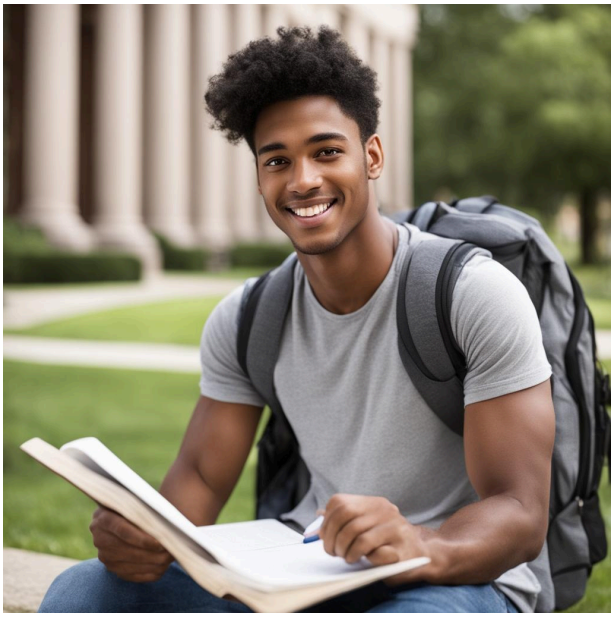
Goals:

- Help students improve academically
- Prevent inappropriate content from being posted

Pain points

- She does not want to lose too much of her free time
- Aware that not everyone is available during her office hours

Benjamin Benson (Tutor)



Benjamin is a senior at SFSU. He excels in his classes and helps out his friends during their study sessions. Benjamin realized the potential to earn some money through tutoring and decided to start at StudyGator. He has sufficient skills in navigating the internet.

Goals:

- Earn Money
- Improve his understanding by teaching
- Help others improve

Pain points:

- He does not know how to advertise his services

3. Use Cases

Use Case 1: John is a new StudyGator user (Student)

John is looking for a tutor to help him prepare for the upcoming organic chemistry midterm. His friend recommended the StudyGator website, which advertised specifically to SFSU students. John uses the site's search and price filter option, entering the specific course name/ID. After browsing through a list of available tutors, John selects Benjamin. John clicks a button to message Benjamin about his services. As John clicks "send message," he is prompted to log in or create an account (lazy registration). After registering and logging in, the message is sent with the content he has written out.

Use case 2: Mary (admin) approves tutor listings.

Mary is a computer science professor proficient in MySQL Workbench, which is why she was hired to administer the StudyGator website. Through Workbench, she checks the listings of users who have filled out the tutor application form. After checking for malicious and inappropriate content, she approves or denies the application.

Use Case 3: Benjamin (aspiring tutor) fills out the tutor listing form and receives approval.

Benjamin wants to become a math tutor for Math100 as a side hustle. After logging into his account, he clicks on the "Apply As Tutor" button and fills out the form. He will have to wait 24 to 48 hours for an administrator to approve his application. The administrator approved his application the next day, so Benjamin can now see that his listing is available for other users to view.

Use Case 4: Benjamin (tutor) receives a tutoring request (message).

Benjamin checks his user dashboard for any new messages from potential students. He sees one message from John, who sent a message containing his contact information (email). Benjamin is able to see which tutor listing the message was sent from and contacts John through third-party services (outside of StudyGator).

4. Main Data Items and Entities

- **User type**
    - **Unregistered user—**The user is not registered to the website. They can browse/search tutor listings, register for an account, or log into an existing account. If they want to schedule a meeting with a tutor, they will be prompted to register/log in.
    - **Registered user** - Browse and message a tutor—inherits permissions of unregistered users. Registered users can fill out a form and attach their resume/CV to create a tutor listing, although it requires admin approval. They can view their messages, listings, and ratings. They can also delete a listing in their dashboard.
    - **Admin -** Has permissions for both registered and non-registered users. Can approve or deny all tutor applications and listings (videos, CV, tutor's bio). They can edit user's accounts. Moderate user-generated content like reviews and ratings.
- **Tutor Listings** - Information inputted by the tutor about their services.
    - Tutor image
    - Tutor name
    - Tutor CV in PDF format (optional)
    - Tutor video (optional)
    - Listing title
    - Availability/time slots (optional)
    - Tutor description/information
    - SFSU-specific Subjects/Classes
    - Pricing
    - Reviews/Ratings
- **Dashboard**
    - Profile
    - Notifications
    - User listings

    - Booked sessions
    - Session requests
    - View messages
- **User Login Information** - Created during registration
    - SFSU-specific email
    - Password
    - Name
- **Search categories -** These are categories that a user can use to filter listings.
    - Subjects
    - Classes listed by tutors
    - Tutor name/course name

5. High-Level Functional Requirements

Unregistered users

1. Unregistered users shall be able to browse tutor listings.
2. Unregistered users shall be able to search and filter listings by subject, rating, and availability.
3. Unregistered users shall be able to register an account using an SFSU email.
4. Unregistered users shall be able to view tutor profiles, bios, and availability.

Registered users

5. Registered users shall inherit all privileges of unregistered users.
6. Registered users shall be able to apply to become a tutor.
7. Registered users shall be able to message a tutor for their services from the listing.
8. Registered users shall be able to manage their account information (name, email, password, and account deletion).
9. Registered users shall be able to book tutoring sessions based on tutor availability.
10. Registered users shall receive notifications for messages or cancellations.
11. Registered users shall be able to leave reviews and ratings for tutors after sessions.
12. Registered users shall be able to reschedule or cancel a tutoring session.
13. Registered users shall be able to upload their CV/Resume.
14. Registered users shall be able to edit their tutor profiles (subjects, courses, availability, or bio)
15. Registered users shall be able to upload their bio videos.

Admin

16. Admin users shall inherit all privileges registered users
17. Admin users shall be required to verify all tutor listings before they are posted.
18. Admin users shall be able to verify the prospecting tutor’s application.
19. Admin users shall be able to remove users or listings from the website.
20. Admin users shall be able to view and manage all user data.

6. 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 render 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 simple to use and intuitive
10. The application shall follow established architectural patterns
11. Application code and its repository shall be simple to inspect and maintain
12. Google Analytics shall be used
13. No email clients shall be allowed. Interested users can only message to sellers via in-site messaging. One round of messaging (from user to seller) is enough for this application
14. Pay functionality, if any (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 2024. For Demonstration Only" at the top of the WWW page Nav bar. (Important so as to not confuse this with a real application).

7. Competitive analysis (functions/features only, not business or marketing)

Feature	Our website **StudyGator**	Chegg Tutor	Tutor.com	Khan Academy	Wyzant
User Profile	++	+	+	+	-
Search Functionality	++	+	+	+	+
Tutor Matching	++	+	++	+	-
Session Scheduling	++	+	+	+	-
Campus focus (SFSU specific-features)	++	-	-	-	-

Feature exists; ++ superior; - does not exist

Our tutoring website offers unique value by focusing on SFSU-specific features, such as matching students and tutors within the SFSU community and integration with the university's scheduling. Competitors like Chegg, Tutors, and Wyzant provide robust matching and search functionality but lack the personalized, university-specific focus that our platform will offer. Additionally, the advanced filtering and customized scheduling features will make it easier for SFSU students to find and book sessions with tutors who have already taken the course, ensuring they receive support tailored to their specific academic needs.

8. High-Level System Architecture and Technologies used

- Deployment cloud servicer we are using:
    - AWS EC2
    - Cloudflare and GoDaddy for custom domain
- Database:
    - AWS RDS and MySQL v8.0.39
- Web Server:
    - Nginx/1.24.0 with PM2 process manager
- Operating System:
    - Ubuntu 24.04
- Frontend frameworks:
    - ReactJS
    - Tailwind CSS
- Backend:
    - Node/Express v20.17.0
- Browsers we will support:
    - Google Chrome
    - Microsoft Edge
- Additional open source resources/APIS:
    - Flowbite for UI components and Google Analytics

9. Use of GenAI tools for Milestone 1

genAI tool	ChatGPT-4o
Task	Grammar check/fix and brainstorming ideas for use cases
Useful rating	Medium

- Prompt: Can you fix the grammar/clarify the following text? [Insert fragment of the milestone one writing]

Although we do not have the prompt's history anymore, we basically had GPT perform grammar/spell check for our writing as well as to get considerable ideas for our use cases. One noticeable help it provided is in the last paragraph of the Executive summary. It helped clarify and get the point across to the reader more concisely and expanded on existing ideas. Alternatively, GPT spewed a lot of random jargon and added unnecessary fluff to the writing, but that was filtered out and removed manually. Overall, it was helpful with grammar/spelling, but it still required a lot of intervention and correction to get the writing quality we desired.

genAI tool	DeepAi
Task	Image Generation
Useful rating	High

We also used AI to generate images for our personae. While the pictures are not perfectly realistic, they serve their purpose well by providing a face for our several personalities. Generating all three images took around five minutes. Overall, they are acceptable and were generated quickly despite their slight uncanniness.

10. Team and Roles

Student Names	Email	Role
Bryan Lee	blee37@sfsu.edu	Team Lead
Min Ye Thway Khaing	mkhaing1@sfsu.edu	Front-end Lead
Kenneth Wen	kwen@sfsu.edu	GitHub Master
Nishi Suratia	nsuratia@sfsu.edu	Back-end Lead

11. Team Lead Checklist

- So far, all team members are fully engaged and attending team sessions when required [**OK**]
- The team found a time slot to meet outside of the class [**OK**]
    - It is a challenge to accommodate everyone's hectic schedule. Some of us work over the weekend, lecture during others' free time, and so on. We have decided to resolve this with flexible online meetings and mandatory email checkpoints every end of Friday.
- 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**]
- The team reviewed class slides on requirements and used cases before drafting Milestone 1 [**OK**]
- The team reviewed non-functional requirements from the “How to start…” document and developed Milestone 1 consistently. [**DONE**]
- The team lead checked the Milestone 1 document for quality, completeness, formatting, and compliance with instructions before the submission [**DONE**]
- The team lead ensured that all team members read the final M1 and agreed/understood it before submission [**DONE**]
- The team shared and discussed the experience with genAI tools among themselves [**DONE**]

- Github is organized as discussed in class (e.g., master branch, development branch, folder for milestone documents, etc.) [**OK**]