SW Engineering CSC648/848 Summer 2022

GatorShare

Milestone 01

Team 01, Section 01

Team Members:

Donna Nguyen, Team Lead | unguyen3@mail.sfsu.edu

Estefanos Kebebew, Back-End Lead

Brianna Soukup, Front-End Lead

Mohamed Toure, Github Master

Brian Nguyen, Database Master

Aleksandr Gusev, Front-End

History:

Submitted:	**July 21, 2022**
Revised:	

Table of Contents

Executive Summary

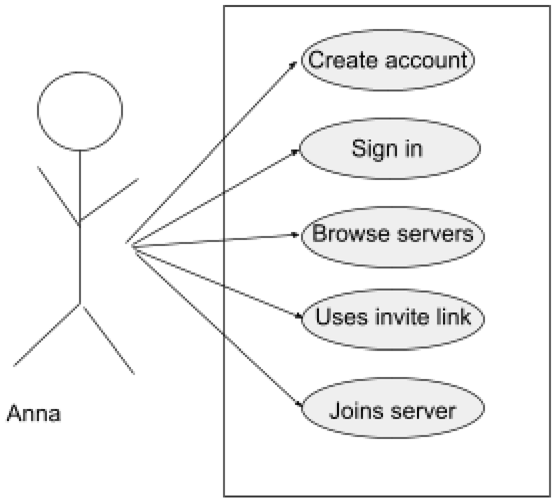
Throughout the years of online learning, San Francisco State University's social networks have become extremely limited for students. Today, students are restricted to only the school's unofficial Discord server and SFSU's homepage. Online learning has shown that many student's social platforms are reduced to Zoom calls, iLearn discussions and casual conversations on Discord channels. We see many students aren't as sociable or approachable due to online constraints.

Our motive is to create a platform where all students are welcomed to communicate with one another and to openly share their thoughts, hobbies, creative skills and social events. Our goal is to build a stronger community within SFSU's online medium by integrating everything into one. With our application, students can communicate with one another and aren't restricted to a few Discord channels. Students can post articles and essays to share their thoughts, post their art, photography and films for recognition and feedback, share club information and meetup times to the public, share Discord servers, and even find tutoring services— all available in one web application. Although our target group is aimed towards students, our platform welcomes faculty and alumni as well. Students do not need to roam around SFSU's website to search for a link, nor do they need to search for different Discord channels in hopes to find an event post. Our goal is to allow a platform where every student can welcome one another, with a focus on sharing content and building greater connections.

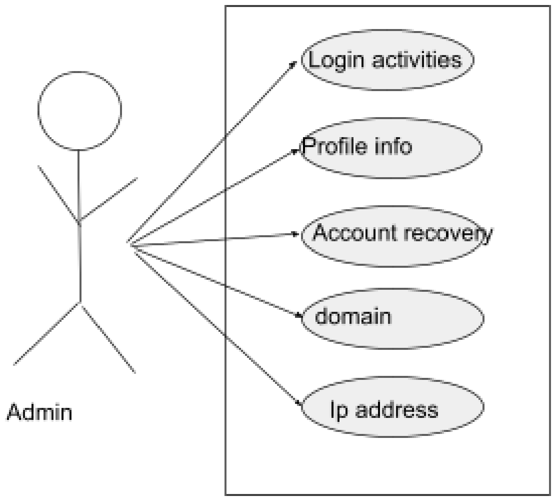
Our system is unique in a sense where nothing is scattered in several different channels or links. We want to create a community where students can share their ideas and receive positive exposure all in one platform.

Main Use Cases

Use Case Title	Finding a Discord Server
Actor:	Registered User - Student
Description:	Anna is a first-year student at SFSU. She wants to join a Discord server and make some new friends. She uses her school email to create an account and log in to our website. Anna notices that she may either search for a specific Discord server with the search bar, or explore all of the servers that have been posted by other registered users. Anna chooses to browse all of the available servers, looking at which servers interest her. After finding a Discord server she likes, Anna clicks the given invite link to the server. There, she is connected to the Discord server and is able to make new friends.

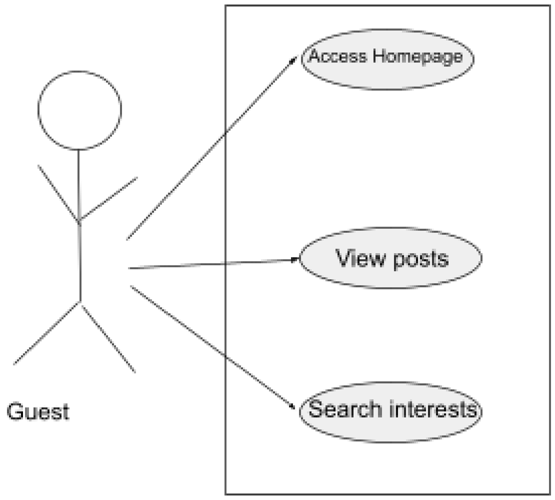


Use Case Title	Deleting Suspicious Activity
Actor:	Registered User - Admin
Description:	When logging in, the Admin are able to view the entire platform. They are allowed to see the login activities of the user. Such as changes of passwords, information, and account recovery. They can also review reports of malicious activities. The report is the composition of domain, login date and time, and Ip address; it can be the user's physical location, proxy server or virtual private network.



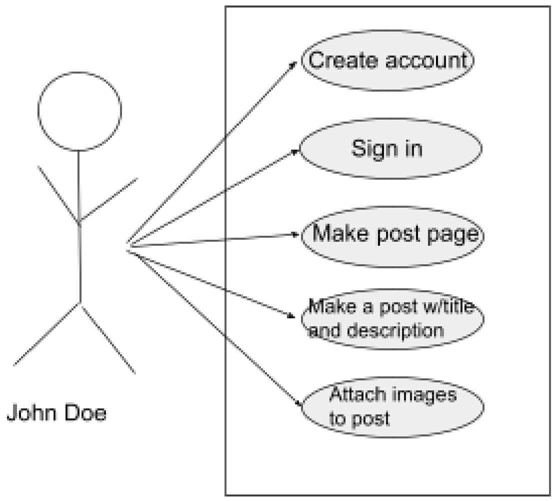
Use Case Title:	Guest
Actors	Unregistered User
Description	A user who has not registered an account or is accessing the page while logged out will be considered a Guest. A guest, when accessing our application, can only view posts. The guest will have full access to view every page within the platform, meaning that they can browse through all

	posts, view club events, access articles, etc. This user can also search for his/her interests. The only limitation an unregistered user has is that they cannot post to the platform, they cannot post comments, they cannot reply to other comments, and they cannot like/dislike any post. If the unregistered user wants to make a post, they must sign up and register an account.

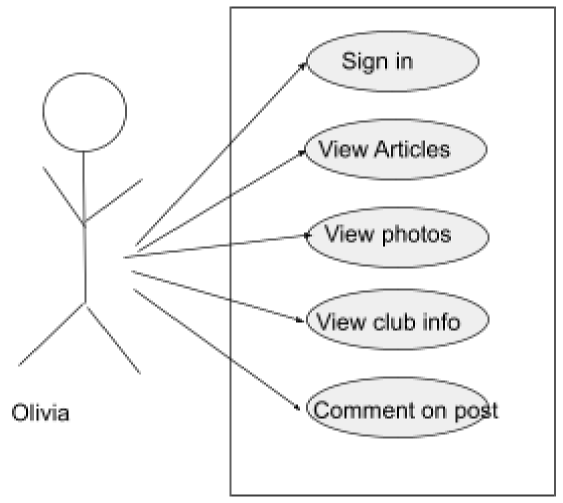


Use Case Title:	Posting
Actors	Registered User - Student
Description	An SFSU student, John Doe, is a Visual Communications major. Throughout his semesters at SFSU, he has built an extensive portfolio of his digital artwork. As John progresses in his academic career, he has decided that he would like to share his work for both publicity and feedback. John stumbles upon GatorShare and sees that other students are posting their original work as well. John decides to register an account, navigate to student posts, and attach several images of his digital artwork. John is

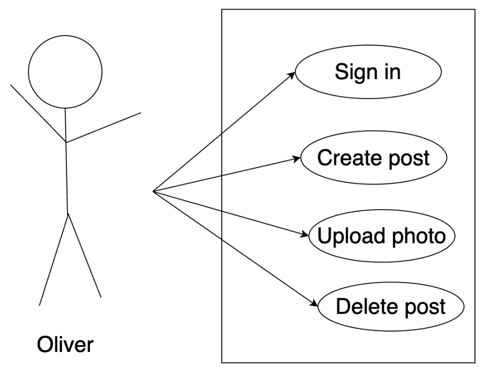
	required to post a title, post an author name, attach at least one image, and add a small description.



Use Case Title:	Commenting
Actors	Registered User - Student
Description	An SFSU student, Olivia, is a registered user on GatorShare. As she is browsing throughout the website, she stumbles upon several student-written articles, original photography posted by her college peers, and even club flyovers about upcoming events. A student-written article talking about a new restaurant opening at Stonestown Galleria captures Olivia's attention. Olivia, being a registered user, decides that she wants to leave a comment on the post. She scrolls to the bottom of the article, and is able to see a comment box under the post. Here, she leaves a comment.



Use Case Title:	Deleting Post
Actors	Registered User - Student
Description	Oliver is a SFSU student and recently went on a hike with his roommate. He decides to upload a picture he took from Mount Diablo. His roommate, when logging into GatorShare saw Oliver's post and wasn't happy about the submission. Oliver signs back into his account, and is able to delete his post.



List of Main Data Items and Entities

User: Any person who uses the web application.

- **Unregistered User:** A User who has signed up or logged into the application. This User is allowed to browse and search through the website, but cannot post or comment on the platform.
- **Registered User:** A User who has signed up or logged into the application. This User must sign up via their full name, school email (to verify SFSU student status), password, and accept our application’s terms of service. This User is allowed to fully utilize the platform (besides Admin privileges). This User can browse and search through the website. This User can also post, comment, view messages, and like/dislike posts on the platform.
- **Admin:** Admin Users are solely restricted to the creators of the web application. These Users are able to fully utilize the platform and can delete Posts, Comments, and Registered Users.

Post: A submission made by a Registered User. A post is composed of a title, text box, and optional media content. A Post can be plain text used as an article, an uploaded image to promote a digital artwork or event flyer, or a discussion post.

Comment: A Registered User can click on another Registered User's post and comment underneath the Registered User's post. A comment would be text only.

Message: A Registered User can message another Registered User.

Initial List of Functional Requirements

Website:

1. Search Bar shall be present throughout all pages.
2. Navigation Bar shall be accessible for all Users.
3. Navigation Bar shall have a Home page.
4. Navigation Bar shall have a Login page.
5. Navigation Bar shall have a Sign Up page.
6. Navigation Bar shall have an About Me page.
7. Footer shall be present throughout all pages.
8. Logo shall be present throughout all pages.
   - When the Logo is clicked, it shall redirect the User to the Home page.
9. Posts shall be public for all Users.
10. Comments shall be public for all Users.
11. Images shall be accessible for all Users.

Requirements for Unregistered Users:

12. Unregistered Users shall not be able to add a Post on the platform.
13. Unregistered Users shall not be able to add Comments to a Post.
14. Unregistered Users shall not be able to receive notifications.
15. Unregistered Users shall not be able to like/dislike a Post.
16. Unregistered Users shall not be able to like/dislike a Comment.
17. Unregistered Users shall not be able to send Messages.
18. Unregistered Users shall not be able to communicate with Registered Users or Admin.

Requirements for Registered Users:

19. Registered Users shall be able to add a Post on the platform.
20. Registered Users shall be able to edit their Post.
21. Registered Users shall be able to delete their Post.
22. Registered Users shall be able to add Comments on another Registered User's Posts.
23. Registered Users shall be able to edit their Comments.
24. Registered Users shall be able to delete their Comments.
25. Registered Users shall be able to receive notifications from other Registered Users.
26. Registered Users shall be able to send messages.
27. Registered Users shall be able to receive messages.
28. Registered Users shall be able to delete messages.
29. Registered Users shall be able to like/dislike a Post.
30. Registered Users shall be able to like/dislike a Comment.

Requirements for Posts:

31. Posts shall have a title.
32. Posts shall have a description.
33. Posts shall not have a character limit.
34. Posts shall have an image submission field.
35. Images shall only be accepted in jpg, png, or jpeg format.
36. Posts shall display the Registered User who created the post.
37. Posts shall display the time of the submission.
38. Posts shall include a comment box.

39. Posts shall include a submit button.

Requirements for Comments:

40. Comments shall not be empty.
41. Comments shall display the username of the Registered User who created the comment.
42. Comments shall display the time of the comment submission.
43. Comments shall have a text limit of 500 characters.

Requirements for Admin:

44. Admin shall be able to delete other Users.
45. Admin shall be able to delete other Posts.
46. Admin shall be able to delete other Comments.
47. Admin shall be able to Post.
48. Admin shall be able to edit their Post.
49. Admin shall be able to delete their Post.
50. Admin shall be able to Comment.
51. Admin shall be able to edit their comments.
52. Admin shall be able to delete their comments.
53. Admin shall be able to send messages.
54. Admin shall be able to receive messages.
55. Admin shall be able to delete messages.
56. Admin shall be able to like/dislike a Post.
57. Admin shall be able to like/dislike a Comment.
58. Admin Users shall be able to receive notifications from Registered Users.
59. Admin users shall be able to post and edit the “About Me” section of the web application.

List of Non-Functional Requirements

Optimization & Performance:

1. The website application language shall be in English.
2. The website application shall be optimized for desktop/laptop usage.
3. The website application shall be optimized for selected browsers such as Safari, Google Chrome, and Mozilla Firefox.
4. The application will upload User's posts and comments in real time.
5. The website application shall display posted media in full width of the screen
6. Images posted on the application shall be in the format of jpg, png, or jpeg only.
7. The website application shall be deployed from the team's Amazon Web Services account.
8. The website application shall store the Users data in the team's MySQL database.
9. User information shall not be duplicated in the database.

Security:

10. Users data shall be securely stored in the backend database.
11. Passwords shall be protected.
12. Passwords shall not be stored in the database as plain text.
13. When an Unregistered User signs up:
    - Passwords shall be at least 8 characters.
    - Passwords shall require at least one number.
    - Passwords shall require at least one special character.

Competitive Analysis

Features	CourseHero	Chegg	SFSU Discord	SFSU Website	SFSU iLearn	SFSU Reddit	GatorShare
Search/Filter	+	+	+	++	-	+	+
User Profiles	++	++	++	++	++	+	+
Like/Dislike	+	++	+	-	-	++	+
Messaging	-	-	+	-	-	+	+
Commenting	-	+	+	+	+	+	+
Online Status	-	-	+	-	-	+	-
Notifications	+	+	+	-	++	+	+
Posting	+	+	+	+	+	+	+

(++) Superior
(+) Same
(-) Does not exist

Summary:

Despite GatorShare being average overall compared to its competitors, our application is the only platform that aims to integrate everything into one. When assessing the chart, we can observe that some of our competitors may be superior to our application, or just about the same. When it comes to searching and filtering, GatorShare is the same as their competitors but serves different information. GatorShare allows users to search posts in order to find tutors, homework answers, events, discussion posts, etc. This can be seen as advantageous simply because CourseHero and Chegg strictly adheres to academic-related content, while Discord only filters comments and users from specific servers, and Reddit may have spam and unsupervised posts, and random discussions unrelated to SFSU. GatorShare is monitored by their Admins, where everything must be related to SFSU and students have a free range to start discussions, post events and flyers, and also search for tutors/get homework answers. SFSU's website has a search engine far more superior than GatorShare simply because of their implementation of Google's search engine, something we are not utilizing. It is worth noting that many of our other competitors are either far more superior with functional user profiles than GatorShare, or about the same. Our application does not have an emphasis on the functionality of user profiles, as our main focus is the capabilities of our overall platform instead. We can also see that SFSU's Website and SFSU's iLearn lacks a like/dislike feature, and our other competitors such as Chegg and Reddit may be more desirable. Our application does not have a system like Chegg, where users are compensated for having likes (on an answer), nor does it have a system like Reddit, where a post may be "hot" or "popular" based on the number of likes. Our like/dislike feature is mainly used as a means to support another User's content. We also pride ourselves in our messaging and commenting features, as our messaging system allows one SFSU user to connect with one another and our commenting system allows SFSU's to share compliments, give input, and admire other students for their original work. Overall, although our application may be seen as average, this should not be seen as a disadvantage, since GatorShare has a different goal in mind. Our platform aims to create both an online learning platform and a social website for our peers. It is made by students and is catered to students.

High-Level System Architecture and Technologies Used

Software: Postman, Swagger Editor

Programming Language: Java, Javascript, SQL

Markup languages: HTML, CSS, JSON

Database System: MySQL.

Deployment Platform: Amazon Web Services, AWS Amazon Linux (Inferred)

Web Server: Apache Tomcat

Frameworks:

- Front End
  - Bootstrap, React
- Back End
  - Spring, Reddis

APIs:

- Spring Boot

IDE

- Microsoft Visual Studio Code
- Sublime Text
- Eclipse
- Intellij

Checklist

1. Team found a time slot to meet outside of the class

DONE

2. Github master chosen

DONE

3. Team decided and agreed together on using the listed SW tools and deployment server

DONE

4. 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

5. Team lead ensured that all team members read the final M1 and agree/ understand it before submission

DONE

6. Github organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.)

DONE

List of Team Contributions

Donna Nguyen, Team Lead	Drafted the Executive Summary, where the team reviewed. Team Lead finalized the summary alongside M1 Editor in order for the whole team to be on the same page. Helped contribute to main use cases, and reviewed each case in order to make sure each implementation is logical and relevant to the applications purpose. Drafted the list of main data and entities with the help of teammates. Afterwards, Team Lead then reviewed and finalized alongside M1 Editor. Drafted the list of functional and nonfunctional requirements then had the team review each requirement. Team Lead finalized the requirements alongside the M1 editor to make sure each requirement is feasible. Led team meetings and checked in on team members. Kept up with questions, communication, and documentation. Reviewed and finalized high-level system architecture. Summarized the competitive analysis with the help of teammates research and assessment.
Estefanos Kebebew, Back-End Lead	Successfully registered the application's domain from AWS Cloud. Researched where to host the website that AWS provided. Tested several different types of cloud storages such as using amplify, S3 Bucket, and elastic beanstalk. Decided to go with S3 Bucket and used Cloudfront to access the bucket. Used Cloudfront to create SSL certification. Debugged back-end and server-side errors to help the application project run. Implemented AWS RDS and AWS Instance to host the database. Helped teammates connect to the AWS Cloud server. Helped teammates connect to MySQL database. Efficiently fixed bugs related to the backend. Helped with constructing main use case ideas. Reviewed and finalized functional and nonfunctional requirements to make sure all requirements are feasible. Reviewed and finalized high-level system architecture. Researched and contributed to competitive analysis.
Brianna Soukup, Front-End Lead	Contributed to many discussions and ideas in the team's Discord server. Proposed the idea of our web application. Provided ideas for the basic function of our application's purpose. Contributed to main use cases by coming up with main use case ideas. Created use case diagrams for each main use case. Helped come up with, and created the application's logo for our website. Created a shared Figma file for the frontend team to utilize the creation of user flow diagrams and wireframes in the near future. Reviewed and finalized functional and nonfunctional requirements to

	confirm that the requirements are feasible. Researched and contributed to competitive analysis.
Mohamed Toure, Github Master	Designated M1 editor. Moved code in the development branch into the testing branch. Helped test files and finalize code before moving to the master branch. Helped edit M1 documentation. Helped review and finalize executive summary to make sure it's legible. Reviewed main use cases. Researched and contributed to competitive analysis.
Brian Nguyen, Database Master	Helped brainstorm and create a table within MySQL for users with basic information (rough draft). Reviewed and finalized functional and nonfunctional requirements in order to make sure the requirements are feasible and implementable. Contributed to creating data items and entities by coming up with ideas and adding input. Helped review and finalize data items and entities. Helped provide documentation for back-end lead and also provided documentation about the database thus far. Researched and contributed to competitive analysis.
Aleksandr Gusev, Front-End	Created the front end codebase (React App) and created a rough outline of the web application. Created and designed the "About Us" page. Helped sketch a rough draft of the user flow diagram. Contributed to main use cases along with reviewing each use case. Gave input to the Front End Lead about the application's logo design in order to have a presentable logo for display. Reviewed and finalized functional and nonfunctional requirements to confirm the requirements are feasible. Helped manage and maintain the frontend branch of the GitHub repo. Helped with formatting the M1 document. Consistently updated the document's table of contents and reviewed M1 document for errors. Fixed push and overwriting errors on GitHub. Cleaned up repo. Researched and contributed to competitive analysis.