SW Engineering - CSC648/848
Spring 2020

Team 03
Milestone 1

Team Lead - Lionel Wong, lwong7@mail.sfsu.edu
Backend Lead - Jorge Landaverde
Backend Engineer - Gordon Lam
Front End Lead - Mitul Savani
Front End Engineer - Kevin Huang/Jack Kower
Git Master - Mitul Savani

Milestone	Date submitted
Milestone01 Version01	3/5/2020

Table Of Contents

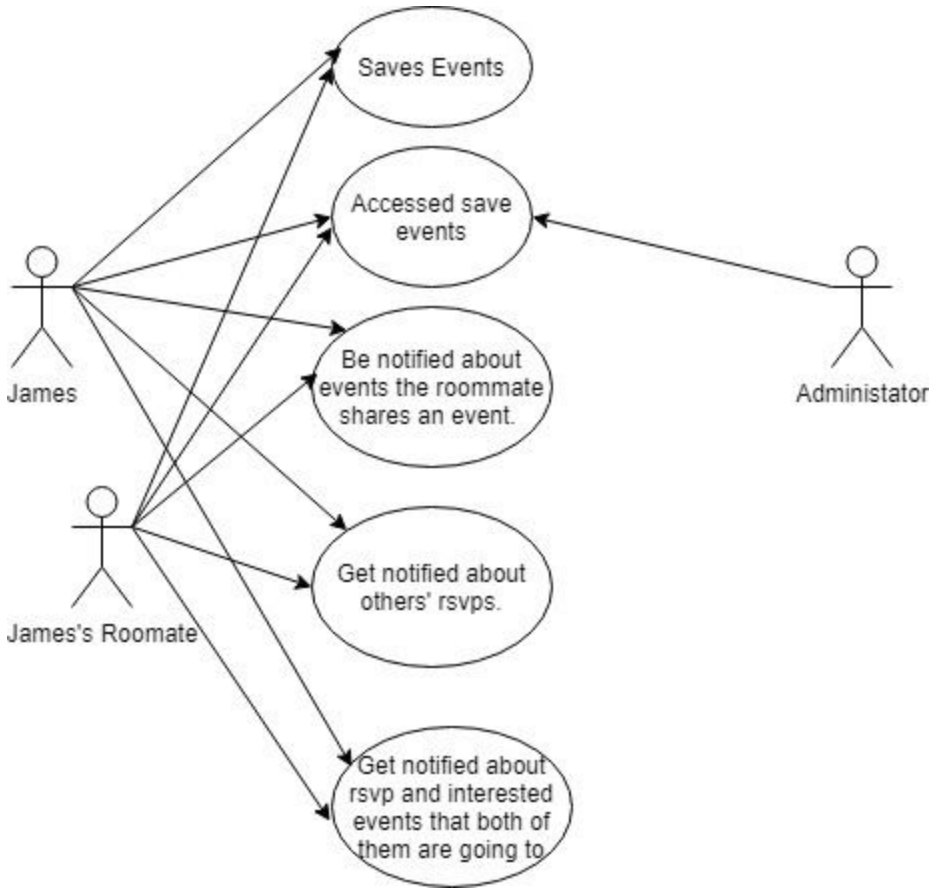
Executive Summary

Unigator is a website dedicated to viewing and organizing San Francisco State University (SFSU) campus events. We aim to provide a platform for SFSU students to easily and clearly browse through university events in one central platform. There are numerous events in SFSU constantly ongoing everyday that don't get noticed and allows funding to be wasted. The current SFSU events' website is clearly lacking in terms of information and functionality. The events we would be displaying are not limited to certain elements, such as education or recreation. Events displayed cover all range of activities, any organization or students that wants to set up an event can use Unigator as a platform to advertise, manage, host or attend events. When anyone wants to know or look up an event at SFSU, Unigator is the place to go.

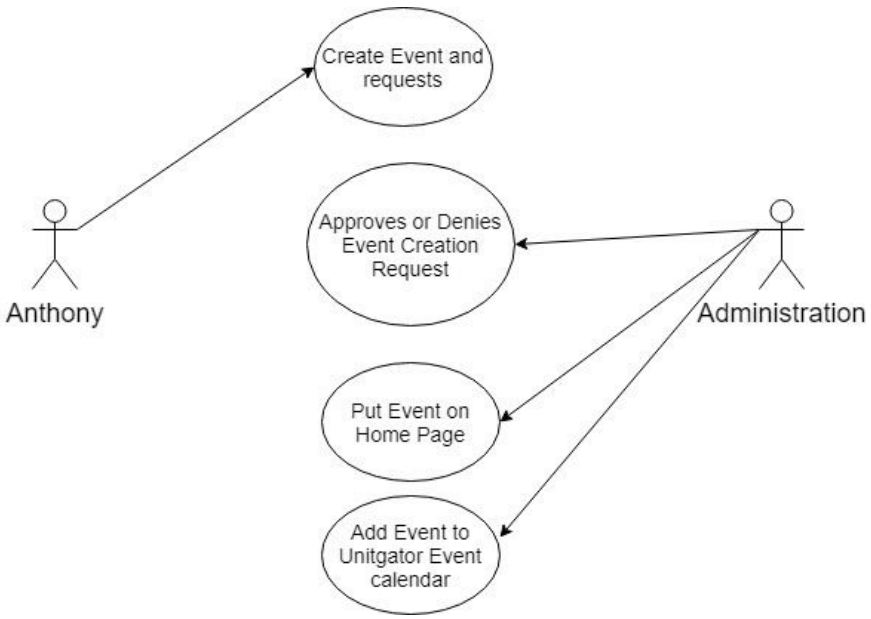
Unigator would take in all the ongoing events at SFSU, whether it's an event created by an organization or by a student. Unigator will prioritize displaying the information of the events in a simple and clear manner. Users would also be able to efficiently manage events and potentially invite or notify others about scheduled events. The events are not tied down to sponsored organization, any scale of event can be posted. This will help the users to clearly identify which events they are interested in. Unigator will be one central platform that will provide all the essential information to make informed decisions. Unigator is aiming to unite students by allowing them to find new experiences to connect with other students.

Main Use Cases

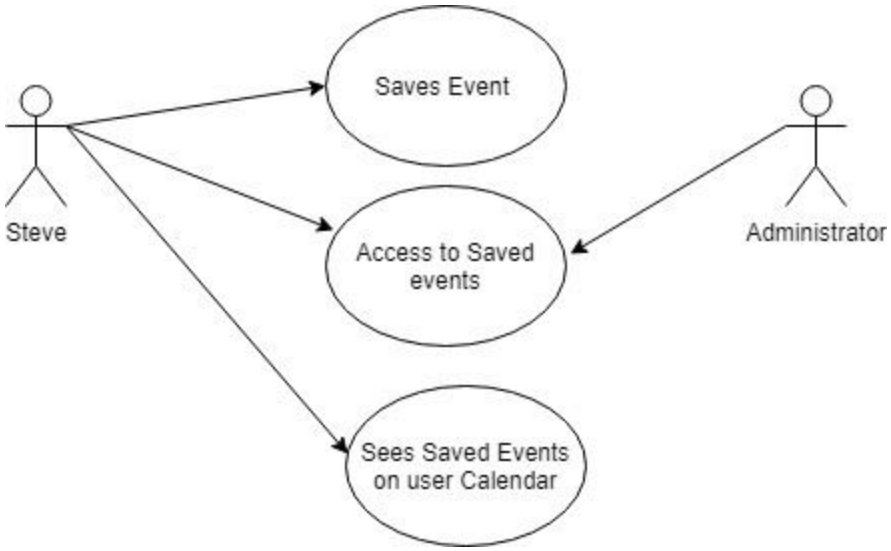
1. James is a 22 year old college student. He is an environmental science major at San Francisco State University. He stays on campus with one other roommate, both James and his roommate are not from around San Francisco and are relatively new to the area and school. James only works part time as a cashier on campus which leaves him some free time. He wanted to explore and get to know things that are happening in the school. James is unsure though where to get info about events that are happening in the school as the school website is cluttered with other information. James's roommate recommended him our online application "Unigator", which easily provides all the essential info and details of any event that are **currently planned or ongoing** in the school. James needs a way to let his roommate know of any events that he is interested in, and his roommate should be able to get **notifications on events** James has **shared with him**, so that they can both plan to attend the event.
   Actors: James, Registered User



2. Anthony is a student organizer at San Francisco State University. He is part of an organization that owns and maintains the arcade on campus. Many people frequent the arcade at school almost everyday, some machines/games in particular have regulars too. They expressed their desire to have a local tournament with the machines, Anthony aggrees and **needs a way to create and let people know of such an event.** The school website doesn't provide a good way to market the event, so Anthony resorted to placing flyers around the arcade to promote the event. Anthony found out about Unigator and decided to use it to **host the event.** Unigator helps promote the event easier by possibly being **featured on the homepage** and **calendar of the site** when **approved by administers.**
   Actors: Anthony, Event Host

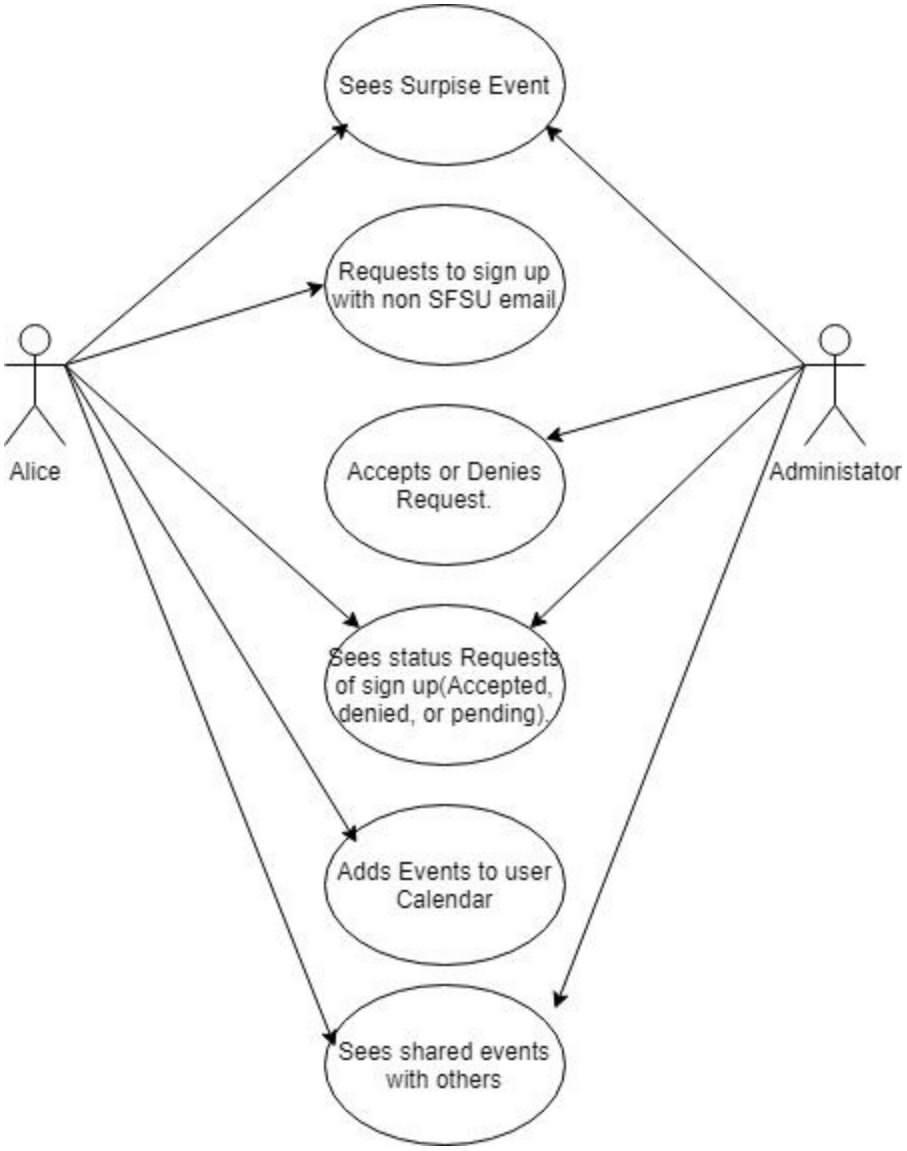


3. Steve is an aspiring software engineer studying in San Francisco State University. Steve needs to find out about possible opportunities to further his career and improve his resume. The school site provides info about most events but Steve is only interested in technology based events, the site doesn't have a **way to filter the events.** A friend recommended the application Unigator, which allows him to **browse** through events and **filter them by technology category,** and provides Steve with all the **necessary information of events** in San Francisco State University. Steve **wanted to save** all the events that seem interesting to him so he can look over and **organize** later on. Steve found out that he will need to **register an account with his school email** in order to use those features. Steve registered with his school email is able to **save the events** and better organize his schedule. Steve also registered his account with his own **unique username.**
   Actors: Steve, Registered User

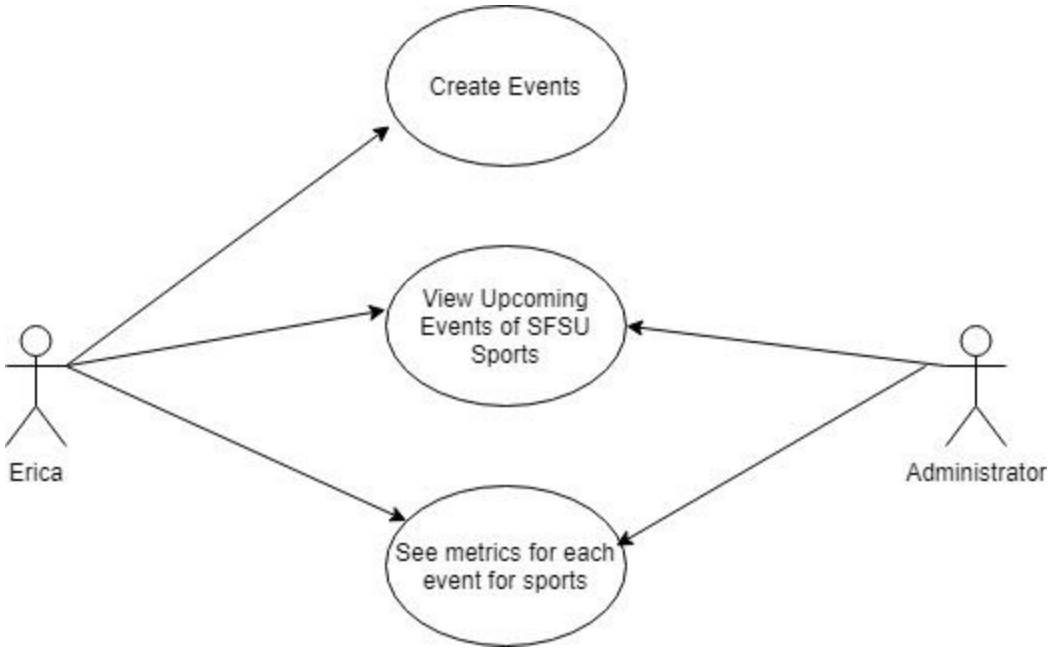


4. Alice is a high school student that plans to attend San Francisco State University in the future. She wants to know more about the school by going to some events in the school, but is unsure of what events she's looking for specifically. The school website is complex, Alice has a hard time navigating the website just to find information about events. She then stumbled upon Unigator, an application about events happening **in San Francisco State University**. Unigator provides a **"surprise" function**, which randomly recommends an event to the user. Alice used this function and got recommended to a social event that she's interested in. She wanted to **save the info** onto a calendar but she needed an account. She is **unable to register an account as it can only be created with the**

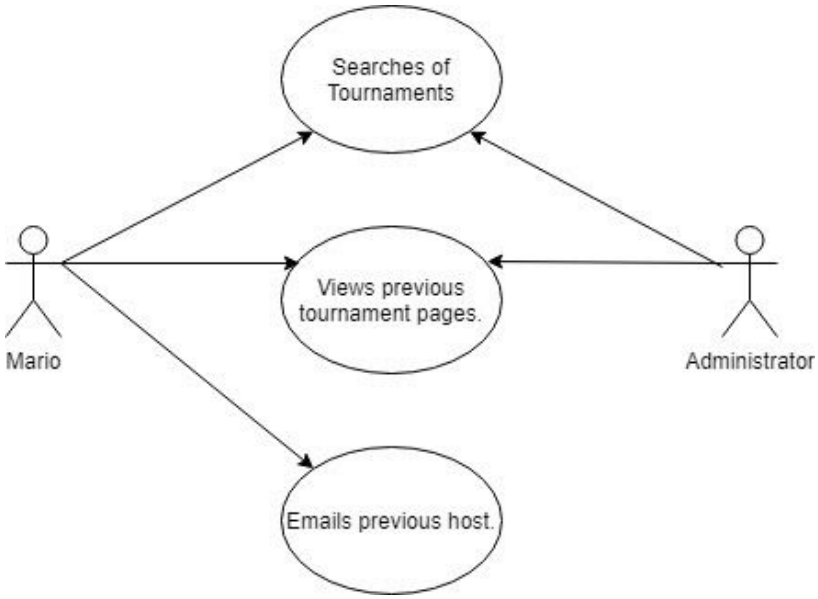
university's email. Therefore, while Alice is able to **browse through events** and learn about it, she has to write down all the relevant info about it. Because Alice is **not a registered user**, she is **not able to share an event's info** and her engagement of it with her social media platform. Alice learns that there are two different ways to become a registered user; she can either create an account with a sfsu email, or she can **submit a request to an administrator** to be able to **register with a personal email.** After being approved by an administrator, Alice is able to create an account without having an sfsu email and is now able to use the **share function** to let her friends and family know what events she is interested in.
Actors: Alice, Unregistered User



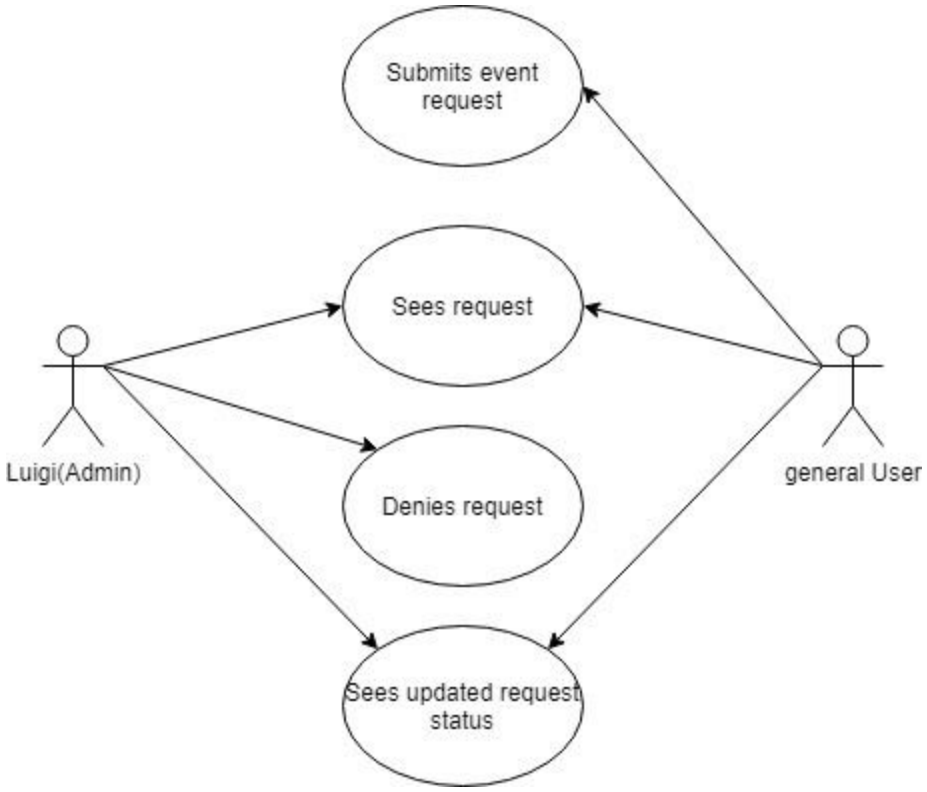
5. Erica is an organizer for the athletic teams in San Francisco State University. She is responsible for entering all of the dates and times for various sporting events taking place on campus. Previously, sporting event schedules are displayed only on the SF State Gators athletics website, but ever since discovering Unigator, Erica has been **posting the events** there as well. The old SF State Gators website only shows the time, date, and location of sporting events. It doesn't offer an option to **reserve tickets online** ahead of time. Tickets are distributed at the gate to sporting facilities only. SFSU students are allowed to attend sporting events for free, so registered student accounts on Unigator can **RSVP** and **reserve a digital ticket** for free ahead of time, eliminating the need to pull out their student ID card at the ticket booth. Unregistered users can still purchase a ticket in person at the event. Erica can use ticket reservation numbers to gauge student interest ahead of time.
   Actors: Erica, Event host



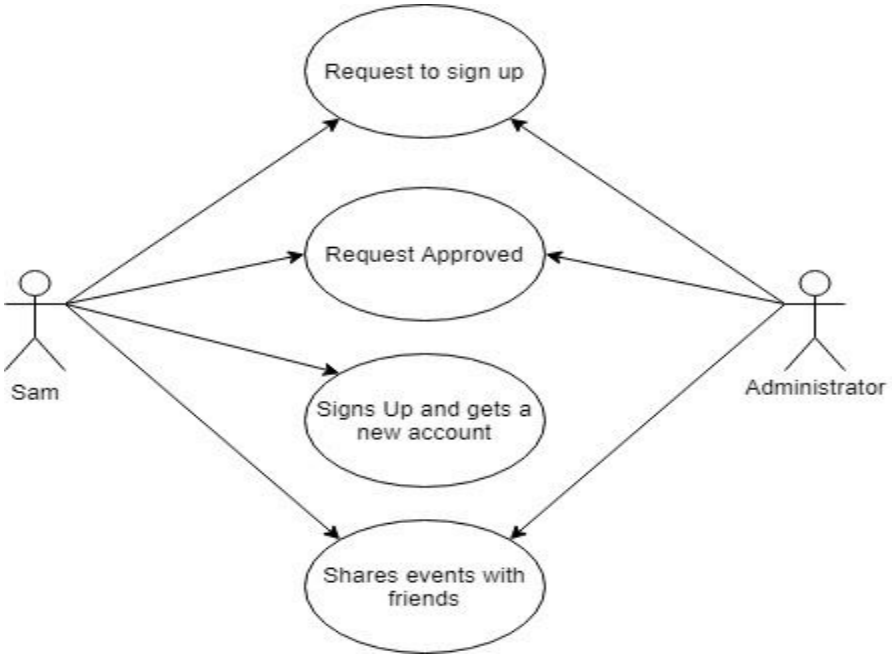
6. Mario is a freshman at San Francisco State University, on his way to class he overhears some other students talking about a Super Smash Bros tournament on campus, a game which he plays and would like to check out himself. Mario goes to unigator and uses the **search bar** to find information about future tournaments, but the only results which **show up are of past tournaments.** Still wanting to get information about the next tournament, Mario **views one of the past tournament's event pages** where he finds the **email** of the previous **event host** and is able to get into contact with them.
   Actors: Mario, Unregistered User



7. Luigi works for Unigator as an **administrator**. All events need to go through Luigi in order to be **approved**. Luigi **logs on**to his account, which has the **administrator privileges**. Luigi goes through the **event approval list** and approves those he sees fit. Luigi came across an event request that was unreasonable and confusing, he **rejected** the proposal and a **notification was sent** with explanation of why it was rejected. After Luigi has done his job, Luigi just casually **browse through events** like any other users. After going through the list, Luigi **logs out** of his admin account.
   Actors: Luigi, Administrator



8. Sam is a student at a local community college. He is interested in cultural events. He sees 2 events he wants to go to with his friends at SFSU. Since Sam doesn't go to SFSU, he can not create an account. So he requests to sign up with a non SFSU email from the administrator. His request gets approved and he is able to share the events he is interested in with his SFSU friends on social media and is able to RSVP to the events.
   Actors: Sam, Unregistered User



List of main data items and entities

Data Entities	Definition
Unregistered User	Users that are not registered with the site, able to browse through events and look at the information, but are unable to utilize some functions of Unigator, such as bookmarking the event and sharing it.
Registered User	Users that are registered with the site with their SFSU email. Able to browse through events and save them into their own personal calendar and share the events with social media. Able to update their status whether they are attending the event.
Events	Activities that are displayed on Unigator that takes place in SFSU.
Event Host	The registered user that is hosting the event, provides all the essential info of their event such as date, time, location and description.
Event Guest	A registered user that declared that they are attending the event. This allows the event host to gauge how much attendees the event might potentially have.
Administrator	A higher privilege user that oversees the site. Able to decide whether an event is to be listed or not. Able to ban users or send warnings.
Approved Background Credentials	Email needs to be verified as a SFSU email.
Event Organizers	An organization that coordinates the event and sets up all the required fields. Makes sure that the location and time is suitable to host the event and communicate with the school if there is any disagreement.
Event Location	The location of a specific event.

Event Time	The time of a specific event.
Event List	A list that contains all of the events currently scheduled or ongoing in SFSU.

Initial list of functional requirements

Unregistered Users

1. Unregistered Users shall be able to sign up with their SFSU email.
2. Unregistered Users shall not be able to sign up without their SFSU email.
3. Unregistered Users shall be able to browse through the event list.
4. Unregistered Users shall be able to see all the essential information about an event.
5. Unregistered Users shall not be able to bookmark/star an event.
6. Unregistered Users shall not be able to share with their social media.
7. Unregistered Users shall not host or create an event.
8. Unregistered Users shall not be able to utilize the calendar function.
9. Unregistered Users shall be able to view past events.
10. Unregistered Users shall be able to make a request to the Administrator.
11. Unregistered Users shall not be able obtain points based on interactions on the site.

Registered Users

12. Registered Users shall be able to browse through the event list.
13. Registered Users shall be able to see all the essential information about the event.
14. Registered Users shall be able set a unique username.
15. Registered Users shall be able to view past events.
16. Registered Users shall be able to report any event.
17. Registered Users shall be able to bookmark/star an event.
18. Registered Users shall be able to share with their social media.
19. Registered Users shall be able to host or create an event.
20. Registered Users shall be able to utilize the personal calendar function.
21. Registered Users shall be able to obtain points based on interactions on the site.

Administrators

22. Administrators shall be able to regulate the site.
23. Administrators shall be able to see all the information about an event.
24. Administrators shall be able to browse through the event proposal list.
25. Administrators shall be able to reject an event proposal.
26. Administrators shall be able to ban a registered user.
27. Administrators shall be able to remove any events.

28. Administrators shall be able to respond to any requests by any user.

Event Host

29. Event Host must be a registered user.
30. Event Host shall be able to edit or remove the event that they are hosting.
31. Event Host shall be able to request any special changes to Administrator.
32. Event Host shall have the same functionality as a registered user.
33. Event Host shall be able to have an Event Organizer to host/coordinate with an event.

List of non-functional requirements

1) Security:
    a) Can only star an event page if you are logged into your account.
    b) You can only create an account with a SFSU email.
    c) Passwords will be encrypted.
    d) Have password requirements for creating an account and resetting password.
2) Audit:
    a) The user may request access to create an account without SFSU email.
    b) The administrator may delete any event from the event page.
    c) The administrator may view all rsvp accounts of any event.
    d) New users will be audited by the administrator.
    e) New Clubs must be approved by the administrator.
3) Performance:
    a) The loading time on the site shall not exceed 3 seconds for any device
    b) The application shall be able to retrieve information from the database in a timely manner
    c) The site shall handle requests asynchronously following a REST format.

4) Capacity:
    a) The total data storage of the web application should not exceed 85% of the data allotted on the server.
    b) The website shall be capable of handling at least 50 users
    c) The website shall be scalable, so that new features can be added easily
5) Reliability:
    a) Downtime for maintenance shall be less than 3 hours per month
    b) Downtime for maintenance shall not affect the site's main functionality
    c) In all cases, users shall be informed of downtime for maintenance, either via an announcement on the ain page, or email

6) Recovery:
    a) In case of a total site failure, the whole site shall be shut down for revision
    b) If the site is broken down, the mean time to recovery shall not exceed one day
    c) User data is the most valuable aspect and priority will be placed on recovering such data in case of total failure

7) Data Integrity:

a) All Data will be backed up every 72 hours.
b) The users are prompted when they wish to delete data.
c) Administrator shall be able to execute a recovery if needed
d) Image sizes shall be restricted to at most 1 megabyte
e) Images shall be uploaded in JPG, JPEG, and PNG formats

8) Compatibility
    a) The site shall be compatible with Chrome browser version #xxxxx
    b) The site shall be compatible with Firefox browser version #xxxx
    c) The site shall be compatible with Safari browser version #xxxx
    d) Third party applications shall not be able to modify any content that may affect the site compatibility
    e) Content shall be able to be ignored by most popular ad-block services
    f) The site shall scale correctly with the screen size of the device its on
    g) The site shall be compatible with any client OS that supports the browsers we have tested it on
    h) The site shall be able to account for any compatibility issues as a result of browser or OS updates
    i) The site shall be compatible to escalate to new databases
9) Conforming with Coding Standards:
    a) Architecture and design standards shall meet all the requirements listed under the High-level system architecture and technologies used section of this document
    b) Design pattern is to be strictly enforced with all aspects of the site
    c) Appropriate documentation must be created for all code that is individually written for future maintenance
    d) Production code shall not have any log or output to the console
    e) All errors must not halt the web application without proper error handling
    f) Only working code that meets all code standards shall be submitted to the main branch of the project repository
    g) Code shall be thoroughly tested and debugged before considered working code
    h) All internal errors and exceptions encountered when writing or modifying code shall be stored in a log
    i) Any error that can affect the site's functionality shall be reported to the user
    j) Errors shall be handled in a way that does not affect the site functionality
    k) The whole production cycle of the site shall be finished at least one week before the delivery date

l) The site shall be tested and debugged as a whole product at least one week before the delivery date
m) The site shall not be launched without all priority one features finished and working
n) All major changed to the application shall be discussed but he team and communicated to the class CTO

10) Look and Feel Standards:
a) The application and it's layout shall look professional
b) The site shall be simple, so that it is usable to a wide range of users, and all previously mentioned parties
c) Targeted users will be the main priority for ensuring usability and readability
d) Elements on screen shall meet the compatibility standards of all supported browsers
e) Elements on screen shall meet the compatibility standards of all supported browsers on mobile devices
f) Elements on screen shall be aesthetically pleasing
g) The site shall be able to work correctly without mouse interaction
h) The site shall be able to work correctly without keyboard interaction
i) Element in screen shall be resized automatically without user interaction when being loaded in all the different platforms supported by the site
j) Application's user interface shall make it easy for user to find what they are looking for
k) Have a loading toggle wheel.
l) Have Links to Unigator social media in footer.
m) Follow the color scheme of SFSU with purple and gold.

11) Internalization / Localization Requirements:
a) The default language of the site shall be English
b) The site shall only allow SFSU students to register accounts
c) The site shall be scalable to incorporate other schools into the user base

12) Website Policies:
a) A link to the policies of this site shall be always visible in all its pages to be accessible by all the parties
b) The site will not ask for nor store any payment information
c) User information shall be kept confidential and secure
d) The website shall allow users to register an account
e) Email verification shall be implemented upon registration
f) User's shall agree to application's privacy policy before using the product

Competitive analysis

Feature/ Company	EventBrite	SFSU University Calendar	Associated Students Events	Gator X	**Unigator**
Strengths	Shows almost all events in SF with important info. Clear and simple to use UI.	Shows events in SFSU, provides all relevant info. Date search filter. Fast response time.	Clear and spacious UI,with relevant info displayed on the homepage. Displays info relevant to students of SFSU.	Lists out Events events by date. Good Events Wall	**Displays all events scheduled or ongoing in San Francisco State University**
Weaknesses	Doesn't show smaller scale events, such as in-school events. Only promotes large events by huge organizations.	Mostly small words, no save features for SFSU account. Only shows events by organizers and not by students. No map and calendar feature.	Not mainly focusing on events, but rather all aspects concerning the school in general. Doesn't show much info and uses google calendar in order to display more info. Too many menus, confusing.	Poor graphics and poor UI. Hard to find and access the application.	**Does not provide info about any event taking place outside of campus.**
Pricing	Depends on the events	Free	Free	Free	**Free**
Social Media	Integration with social media platforms.	None	Only points to their specific social media page, which	None	**Shares the event to social media**

			posts certain info.		**platform**

+ feature exists ++ superior - does not exist	EventBrite	SFSU University Calendar	Associated Students Events	GatorX	**Unigator**
Calendar System	+	-	+	+	**+**
General info	+	+	+	+	**+**
First Glance Info	+	-	-	+	**++**
Checklist/Bookmark	+	-	-	-	**+**
Pictures of Event	+	-	-	+	**+**
Creating an Event	+	+	+	+	**+**
Simple and Easy to use UI	-	-	-	+	**+**
Account Required to use certain features	+	-	-	+	**+**
Map System	+	-	-	+	**+**
Filter System	+	+	-	+	**+**
Event Suggestions	+	-	+	-	**+**
Search Bar/Browse	+	-	+	+	**+**
Social Media Interactiveness	+	-	-	-	**+**
Point System for activeness	-	-	-	-	**++**

The advantages of using our application Unigator over others is that it prioritizes events based on the San Francisco State University campus. Unigator gives the user a central dynamic platform to get all their event needs. Other platforms generally focus on a broad range of events spanning the whole city, but our platform is targeted directly to San Francisco State University students. Students will be able to see all events concerning San Francisco State University on Unigator. Students will also be able to access more features, giving them more dynamic control over their account. Unigator stands out as a unique platform due to its unique point system which would display the activeness of a user, which encourages interaction.

High-level system architecture and technologies used:

- Application's Backend: Node Javascript
- Application's Frontend: React Javascript
- Database: MySQL
- Cloud: Amazon Web Services
- Development Environment: Visual Studio/Webstorm
- OS Architecture: Ubuntu 18.04

Team

Role	Name
Team Lead	Lionel Wong
Github Master	Mitul Savani
Backend Lead	Jorge Landaverde
Backend engineer	Gordon Lam
Frontend Lead	Mitul Savani
Frontend engineer	Kevin Huang/Jack Kower

Checklist

• Team found a time slot to meet outside of the class	**DONE**
• Github master chosen	**DONE**
• Team decided and agreed together on using the listed SW tools and deployment server	**DONE**
• 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**
• Team lead ensured that all team members read the final M1 and agree/ understand it before submission	**DONE**
• Github organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.)	**DONE**