Land Rush

CSCE 482 Capstone Project Report

CHANDLER STANCHFIELD, GIAN RODRIGUEZ, JUSTIN KLINE, SASAN LOTFI, and WYATT GRIFFIN, Texas A&M University, USA

1 ABSTRACT

Fair allocation of land with human comfort and accessibility in mind. That is what we set out to accomplish via this project. The current system for student organizations and their members to receive a portion of land to use during Texas A&M's university events is taxing on Aggies, both physically and mentally. The current process involves camping out days before an event, planting stakes on the ground and extending rope, only for an organization with more manpower to potentially show up and take the area for themselves. What A&M has implemented in Aggie park is plots of land of equal size, and while this removes the need for students to physically stake claim, not all organizations are made equal, and some need more space than others.

This is where Land Rush comes into play, our full-stack web application virtualizes the entire process of land selection, providing a fair, efficient method of having organizations sign up with their members and receive their tailored area of land before an event. By creating a queue of organizations, we entice students to become more active and express interest for an event on time. The system fairly partitions a section allocated by the university depending on the member count of an organization, ensuring a level playing field and holding organizations accountable

Our developed system was a success. We conducted user studies with 25 people, with university, organization, and student results yielding a 100%, 80%, and 100% satisfaction rating, respectively. To ensure our project functioned as planned, we aimed to get as close to 7 square feet per person, which we deemed to be the best fit for outdoor events. Fairness is arbitrary, which is why we asked many members to give us their opinion, dictating whether our algorithm was indeed considered "fair" by those who will use it. The response was an overwhelming yes, with 90% of users saying it was fair, and with a runtime of 0.003 seconds per person allocated, our system successfully decreased the overall processes time from days to seconds.

Our project accomplished what we set out to do from the start, making a fair system that efficiently allocates land dynamically for student organizations attending university events.

2 INTRODUCTION

2.1 Domain Context

The context of this project revolves around one of the most cherished traditions at Texas A&M University. Texas A&M prides itself on its football program, and the student body does too, so much so that student organizations come together to enjoy the practice of tailgating. And although this practice is special, getting ready to enjoy it is not as special of an occasion. The university has allocated sections for use during university events, our platform gives the school the opportunity to highlight these and make it known. These sections contain the land that will be partitioned and assigned to organizations during events. The current system for student organizations to select a plot of land they will tailgate in is outdated, inefficient, and inaccessible by some. Students camp out up to days before the tailgate. The day of, they are forced to rush to a spot and are susceptible to 'strength in numbers', if one organization has more members present at that point in time, they receive a bigger piece of land. Virtualizing this process and creating an organized queue of organizations, and deciding land size by expected member count is an improvement to the existing process and will speed it up while making it more accessible.

2.2 Case Scenarios

Case Scenario 1:
Texas A&M University has acknowledged concerns raised by student organizations regarding the current tailgating location reservation process. Our application offers a streamlined solution, empowering both individual students and official student groups to easily reserve designated areas on campus grounds.

Case Scenario 2:
A 50-member organization desires a seamless tailgating experience for the upcoming Texas A&M vs. UT football game. They anticipate 25 members' attendance and aim to secure a designated space in advance. Our user-friendly application simplifies the process by allowing members to express their desire for a tailgating spot and estimated attendance several days prior. On the day before the event, the application unveils the land division, along with specific location and size details in each member's profile. With allocated space confirmed, the organization can focus on enjoying the pre-game festivities and cheering on the Aggies to victory!

Case Scenario 3:
Seeking deeper engagement in the vibrant Texas A&M University community, a student ponders attending a pre-game tailgate. Our innovative application empowers such individuals by providing comprehensive visibility, seamless integration, and enhanced connection. The student can view which organizations will be present and where, allowing them to immerse themselves at A&M.

2.3 Goals and Constraints

This project aims to develop a full-stack application specifically designed to streamline the tailgating registration process for organizations affiliated with Texas A&M University. The primary focus is to deliver a user-friendly interface that allows organizations to register and secure designated spots within the tailgating area with ease. This is done via API routes that give specific views to each user type, and allow users to perform the tasks they need to, sending any relevant information to our database in the backend. By leveraging automation capabilities of a python algorithm based on average member response time and expected member count, the system will efficiently allocate tailgating locations to each organization, ensuring sufficient space to accommodate their needs and eliminating potential conflicts over spot availability.
However, it's important to acknowledge that this project operates within specific constraints, including time limitations demanding timely completion, scalability requirements for future growth, adherence to stringent security protocols to safeguard data, and compliance with established regulatory standards. Not having access or experience with Geographic Information Software (GIS) limits the accuracy with which we can deliver results, as our mapping is constricted to the Google Maps API, and not exact measurements of the entire sections being partitioned. Additionally, the project scope intentionally excludes aspects such as physical event setup and management, logistical arrangements like equipment rental and parking, and the enforcement of tailgating rules, focusing solely on facilitating the registration process.

2.4 Solution Summary

Imagine a university tailgating scene where organizations effortlessly secure their space, students seamlessly discover their interests, and universities effortlessly manage the entire process. This vision becomes reality with our innovative platform.
Universities leverage predetermined land areas dynamically allocated based on organization requests, ensuring fair and transparent space management. Gone are cumbersome manual processes, replaced by custom profiles for organizations that expedite reservation processes and provide instant confirmations for designated sections with specified metrics. These profiles also showcase organization details to attract students, while the platform facilitates a connection between organizations and potential attendees.
Students enjoy a user-friendly interface to discover participating organizations and their locations, allowing them to register directly and immerse themselves in the campus culture. This automation streamlines the entire process, transforming the tailgating experience from a cumbersome endeavor to a seamless and inclusive community event.
This platform fosters collaboration among stakeholders, creating a win-win for universities, organizations, and students. It's not just about tailgating; it's about connecting the university community in a vibrant and engaging way.

3 RELATED WORK

There have been previous attempts to improve the tailgating experience for those participating. Texas A&M has implemented a tool for individuals and alumni to virtually land rush.[https://aggiepark.ucenter.tamu.edu/tailgating/] This tool serves as a platform for users to pay to reserve a section of land that has been predetermined, with all plots being the same dimensions. This existing method works, but is not inclusive and does not take into account participating parties of varying sizes. Our approach to the problem tackles both of these concerns, with no payment needed, and a platform for any campus recognized organization to request a section of land, and the order in which these requests is taken determines whether there will be sufficient land to accommodate everyone. The size of the organizations will differ, and naturally their sizing needs will differ as well, which is why our algorithm will take these factors into account to return a fair division of land.
Regarding the virtual partitioning of land via an algorithm, there have been studies performed and papers published.
/[https://www.sciencedirect.com/science/article/abs/pii/S0957417417302014] The research shows that an automated process is more successful than a human. The problem that is being solved is one of NP optimization, and that is the issue we are tackling at heart, an optimization problem. The use of genetic algorithms in previous papers is related to the division of land, but takes into account factors other than the ones we are considering, and are rooted more in theory than an implementation of the solution. This will serve as inspiration and can help guide us towards our algorithm, and the solution.

4 REQUIREMENTS

4.1 User Stories and Usage Scenarios

Site Administrator Users:
Users that are responsible for managing the overall website. This includes developing features, handling users profiles, resolving technical issues, and performing site maintenance.

- As a site administrator, I want to modify a university, student organization, and/or individual student's profile from the site for manageability purposes.

University Users:
Users created by Colleges/Universities to host virtual land rush events. These users can handle their affiliated organizations and student accounts, designate land plots for events, and create schedules for events.

- As a university, I need to accept a student organization's request to join my university to ensure only valid organizations are allowed.
- As a university, I need to decline a student organization's request to join my university in case they are not a valid organization.
- As a university, I need to modify an event in case some unforeseen cicrumstances arise and it needs to be moved or cancelled.
- As a university, I need to be able to create and designate areas of land to be used in events to ensure only valid areas are being used.

- As a university, I need to create events taking place on the sections so that Aggies can sign up.

Student Organization Users:
Users who have been verified as officers within approved organizations. These users can verify students within their organization(s) and submit requests to join events with a specified number of interested members.

- As a student organization, I need to accept an individual student's request to join my organization to verify actual members are signed up.
- As a student organization, I need to decline an individual student's request to join my organization if they are not a member.
- As a student organization, I need to remove an individual student from my organization if they are no longer a part of it.
- As a student organization, I need to express my interest for an event so that students can sign up with my organization for said event.
- As a student organization, I need to submit my bid for an event for an opportunity to participate to provide members with the event.
- As a student organization, I need to change an individual student's status to admin of my organization to reflect my organization's status.
- As a student organization, I need to see what events my university is hosting along with information about these events to be up to date regarding them.

Individual Student Users:
Users who have created accounts on the website. They can request to join existing organizations or request to have organizations added to their respective universities as well as signal which organization they want to be a part of for certain events.

- As a student, I need to join a student organization for them to have me registered.
- As a student, I need to leave a student organization if I'm no longer interested.
- As a student, I need to see what events my university is hosting along with information about these events to stay up to date.

General Users:
Users without accounts who can access the website's homepage, register as users, and browse informational pages set up to explain the application.

- As a general user, I need to be able to see the plots of land to rush at my university so I can decide what I want to rush.
- As a university staff member interested in the application, I need to see a page that explains it so that I can decide if I want to use it for my university's land rush.

4.2 Definition of Success

Our definition of success is a fully functioning web application that serves the users and completes the user stories, allowing them to see, update, and delete relevant information to their level of user, and view the result of our back-ends algorithm. The usability testing will yield metrics that we will compare to set standards in order to compare our product to what the expectation for it is. As we have multiple users, all of whom have different desired outcomes, these user journeys will be approached differently and their success will be measured accordingly.

Site Administrator:
For the site administrators, our measure of success will rely on their ability to add and remove all tiers of users.

University:
For the universities that implement our product, our measure of success is that they are able to register within the application and submit their designated land for use.

Student Organization:
Student organizations should be able to register within their university and apply for a portion of the school's designated land with members of the organization that have registered. In the case they receive a part of the plot, they should be able to view where said area is.

Individual Student:
The individual user should be able to register within one or multiple organizations within a university. They will be able to see where the designated portion of land is found.

Algorithm:
The algorithm that partitions the universities designated land will be deemed successful once it dynamically divides the land into a number of sections that are sized relative to the multiple factors taken into account. This information should then be displayed accordingly to the product's users.

5 SYSTEM

5.1 Environmental and Health/Safety Concerns

- **Overcrowding/Mobbing:** If the algorithm creates scenarios where too many individuals are allocated into spaces, this could lead to serious issues if people in the middle of large crowds are harmed and are unable to be pulled out or away from the large group they find themselves within. Additionally, in panic events, a sudden surge from the crowd could lead to accidental trampling or other serious bodily or emotional. Finally, intense crowding could provoke bad actors to intentionally cause disruptions or damage as denser crowds become more easy to victimize.
- **Noise Pollution:** Large group events can result in noise pollution, potentially disrupting nearby ecosystems or communities in the land plotted during major events.
- **Air Pollution:** Increased traffic and activity associated with larger events could lead to higher levels of air pollution in the vicinity, impacting air quality and potentially causing respiratory issues for attendees and nearby residents should there be large amounts of flammables or vehicles in the vicinity, even large dust crowds potentially kicked up.
- **Ground Pollution:** Events often generate significant amounts of waste, including litter, food waste, and disposable items. Inadequate waste management practices can lead to environmental contamination and health hazards.
- **Emergency Access:** Poorly planned land allocation could impede emergency access routes, making it difficult for emergency services to reach attendees in case of medical emergencies, fires, or other unforeseen incidents.

5.2 Social, Political, and Ethical Concerns

- **Equity:** There may be concerns regarding the fairness of land allocation, especially if certain groups consistently receive more desirable spaces due to factors like size or access to the website during rush events. For larger groups, they may obtain too much land, pushing smaller groups out and invalidating the main intent of the algorithm, or in cases where the land granted to them is too small, they may feel they are being unfairly punished due to their size. The system must be equitable enough to account for these concerns while still maintaining a level of equality of outcome. Along with these fairness concerns, virtualizing a process that may be deemed by some as an integral part of Aggie tradition could ruffle some feathers, but the big picture is a net positive one.
- **Accessibility:** The project should ensure accessibility for all student organizations, including smaller or less-established groups, to avoid reinforcing existing power dynamics. Additionally, people with less access to the platform than others should be accounted for, allowing them to take part in the system despite certain limitations that invalidate the purpose of a more accessible system such as ours.
- **Community Impact:** The project's implementation could impact existing social dynamics and traditions surrounding tailgating, requiring careful consideration of community input and concerns. If the system defeats the purpose of an event like this, there would be no reason for anyone to want to use it. A large student body desire to switch from an old tradition

to a new method is not necessarily something that can be stopped, but for those passionate about their schools who see this as a detriment to the spirit of the event, it would undoubtedly create a great deal of concern.

5.3 Manufacturability, Sustainability, and Economics

- **Manufacturability:** While not directly applicable, the efficiency and scalability of the algorithm's implementation could be go along with manufacturing. Where ease of production and assembly are best for efficient operations, the design and implementation of the land allocation algorithm should prioritize simplicity and scalability. If the algorithm can easily accommodate varying demands and scale up to handle larger events or more complex scenarios, we can streamline the process of land allocation and enhance overall efficiency for many use-cases.
- **Sustainability:** Beyond environmental sustainability, which involves minimizing the project's pollution and other impacts on local ecology, sustainability also encompasses economic and social dimensions. By adopting sustainable practices in land allocation, such as optimizing space utilization and minimizing waste, we can contribute to a more environmentally friendly and event planning process that also factors in economic benefit. Additionally, promoting social sustainability through equitable land distribution creates a sense of inclusion and cohesion, ensuring that the benefits of the project are shared equitably among all who take part.
- **Economics:** By optimizing the allocation of land resources, we can generate cost savings for both student organizations and the university. Efficient land allocation reduces the need for excess space and resources, minimizing operational costs associated with event planning and management. Ultimately, by maximizing economic efficiency, the land allocation project can create real benefits for all involved, contributing to the overall future prospects of the university or any other organization implementing this algorithm.

6 DESIGN EXPLORATION

6.1 Comparison of Potential Solutions

The following is a list of three potential applicable solutions to the problem of a lack of viable and fair methods to allot land to student organizations for on-campus tailgating events.

Solution 1: Live Jockeying

This method would involve a live website which dynamically allocates organization space in real time, simulating a true land rush. Colleges would assign a starting time for a designated land plot to open for rushing, and organizations would have time to map out desired plots. Once the site opens, organization members signed in on the website would click locations within the plot that have been pre-divided to create "blobs" which grow and shrink as organizations begin to compact within the plot. These blobs would grow until a capacity threshold designated by the plot size was reached, and blobs would be subsequently partitioned up across the map to fit the space, rejecting blobs under a certain volume and reapportioning land from there via an algorithmic process.

Solution 2: Community Algorithmic Allocation

Similarly to solution one, colleges would designate land to be rushed prior to a game. However, this solution limits individual organization member participation in the process down to simply certifying that they are members within an organization so that organization member counts can be tallied. This would account for organizations with member overlap. Furthermore, this method would involve creating a platform to facilitate collaboration between university organizations prior to a reservation request phase. The organization leads would have the capacity to co-conspire and agree upon ideal land mappings before submitting group bids when the virtual rush begins, queuing bids until a maximum threshold is reached before generating an optimized layout via algorithm to fit these groups together as larger units as opposed to individual ones for more agreeable space-sharing. Groups would only be capable of reserving space for the maximum number of students who are part of the organizations within.

Solution 3: University-controlled allocation

This solution revolved around the university having more control regarding individual plots of land. The school would

select the size and shape of each plot, and then assign organizations themselves. This would mean more work for the university and a potentially less fair outcome for the organizations and students involved.

The selection criteria for the optimal solution primarily centers around perceived fairness and optimization of spatial allocation. This goes against the first option on the grounds that it provides far too much chance for proper optimization and leads to a low perceived fairness for larger organizations. Allowing individual users to rush would end with a large amount of organizations making it to the plotted lands but with very little in the way of space depending upon the ratio of participants to organizations. The lower the ratio, the less satisfied all organizations are likely to be. The criterion are also lacking for the group selection option. This heightens the chance of exclusion as organizations plan and strategize with one another prior to games to grab larger portions of land. This would, in effect, lead to groups banding together and pushing out those who go in alone, or, alternatively, large groups being rejected as their sizeable numbers are too great to be allotted based on their position in the queue. This leaves us with the final option, individual allocation. This allows for each organization to start on level ground. Larger organizations run the risk of pushing other bids out as the total capacity is closed in on, but likewise, they're just as vulnerable to being rejected for coming in too late. Individual selection, thus, is the option we have chosen to move forward with.

6.2 Lo-Fi Prototyping

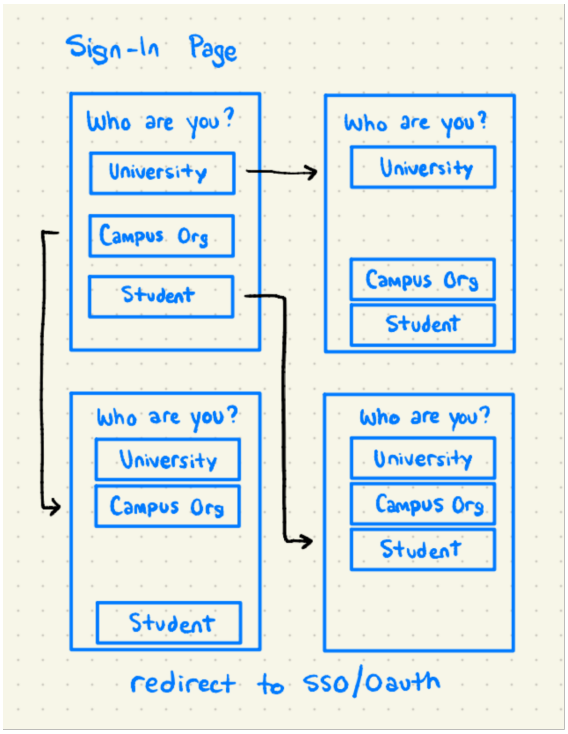

Fig. 1. Sign-In Page

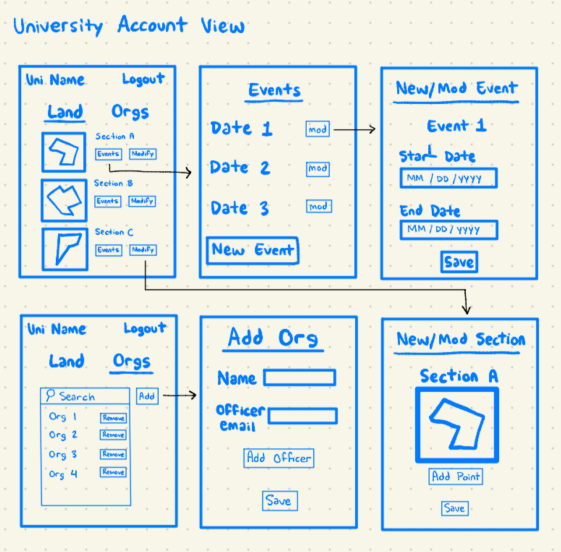

Fig. 2. University View

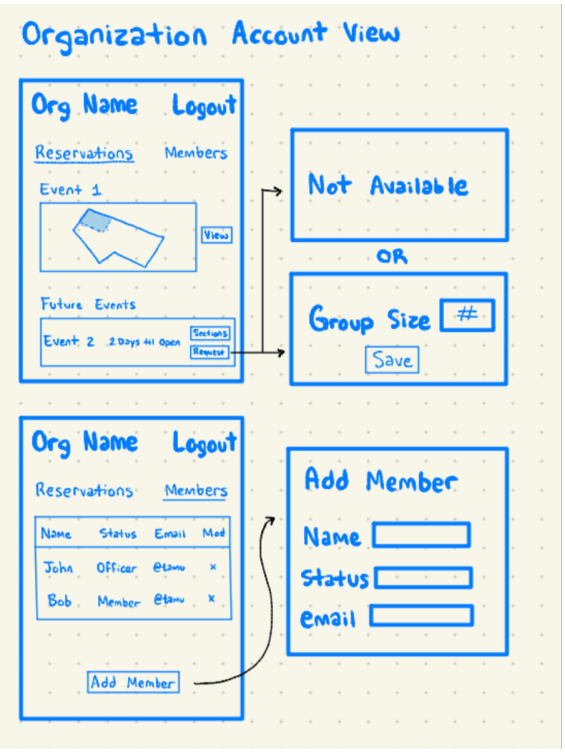

Fig. 3. Organization View

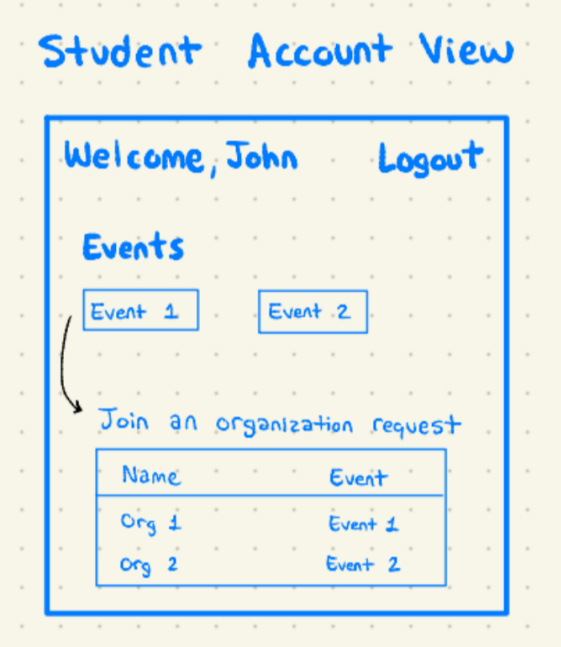

Fig. 4. Student View

Fig. 5. UI Sketches

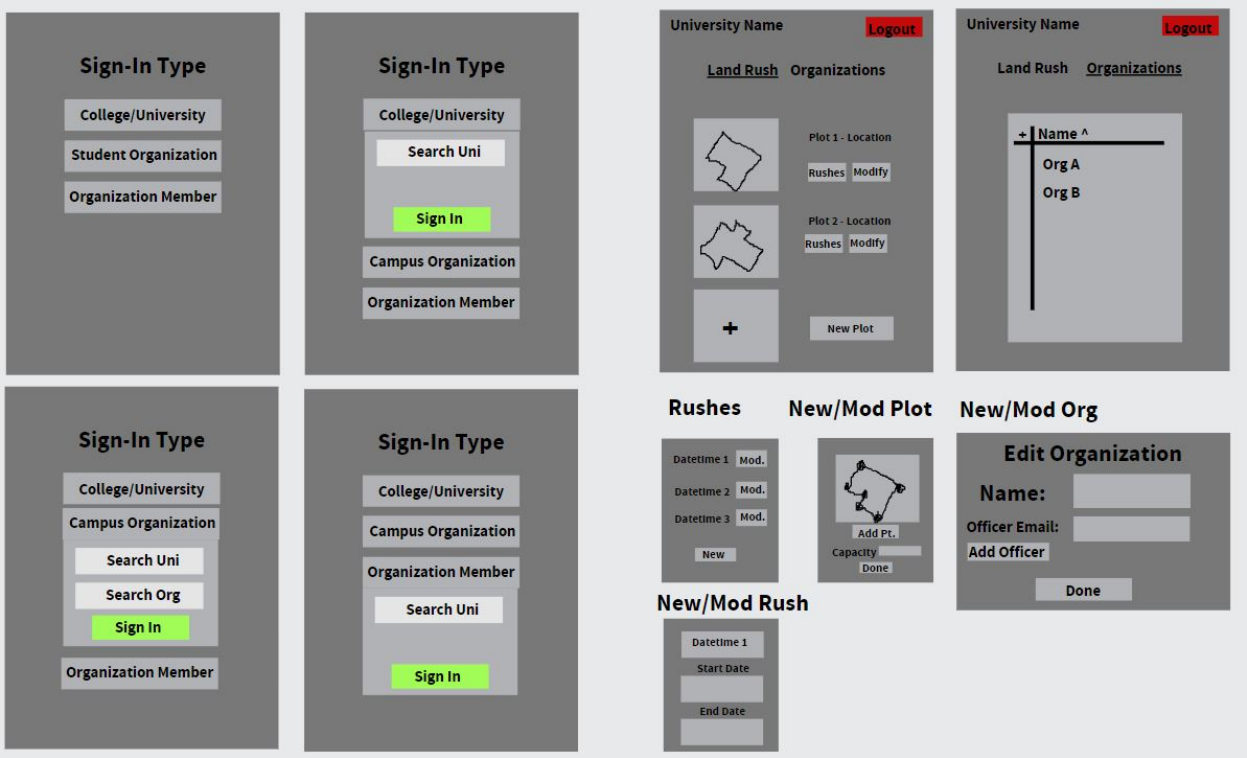

Fig. 6. Wireframe Mockup

6.3 Pilot Studies

6.4 Diversity, Inclusion, Equity, and Accessibility Considerations

Our design is, by nature, an attempt to make the process of rushing a more accessible exercise. Instead of leaving the process to those who are capable of running across a field to stake a claim, perhaps to the exclusion of many students, the opportunity to secure a plot of land for tailgating becomes an activity that any individual can take. This in turn presents opportunities for a more diverse student tailgating crowd as smaller organizations are offered better odds at joining their peers. Likewise, the process becomes more equitable as it places all organizations at the same starting point.

7 SYSTEM DESIGN

7.1 Functional Design

Fig. 7. Level-0 Diagram

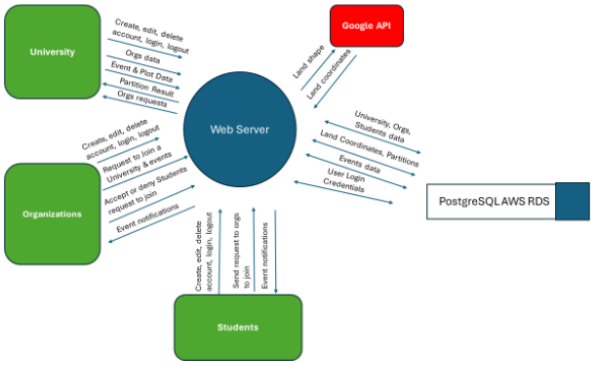

Fig. 8. Level-1 Diagram

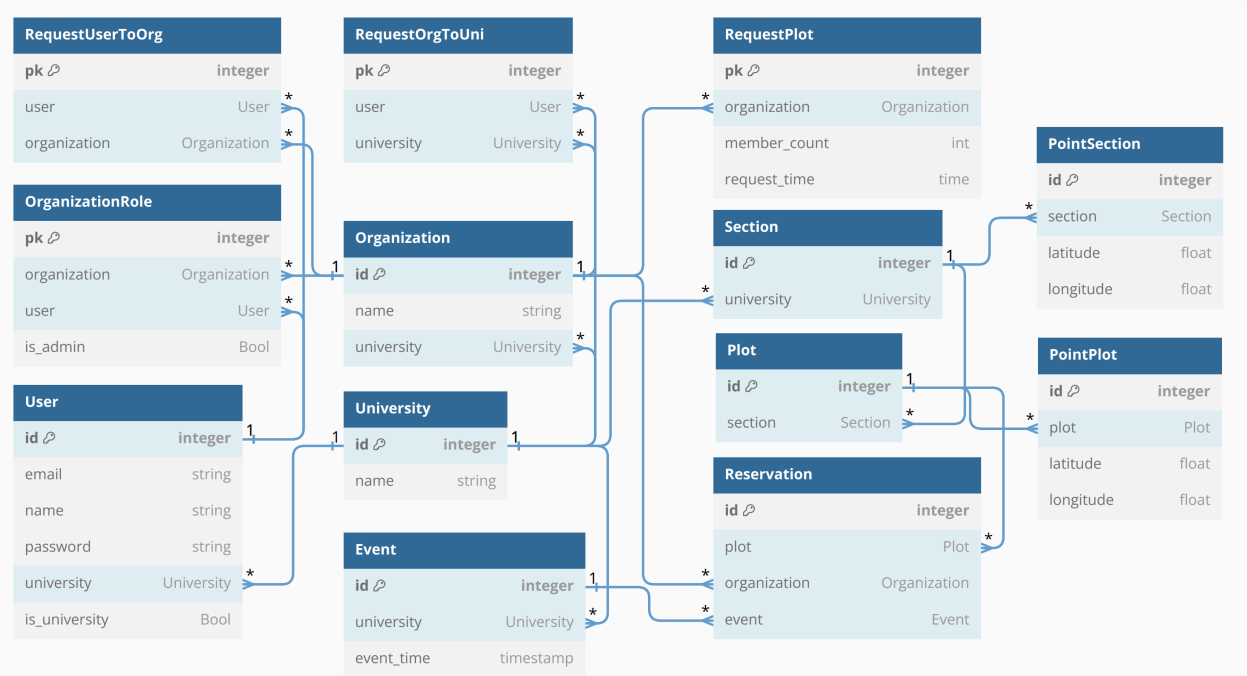

Fig. 9. System Database Schema

7.2 Module Specifications

Student

- Register (must be approved by organization)
  - Select university
- Login
  - JWT (java web token) given to users based on organization and university.
- User Functionality
  - Request to join Org(s)
  - Leave Org
- Delete Account

Organization

- Register
  - Select university (must be approved by university)
- Login
  - JWT (java web token) given to users based on org and university.
- Organization functionality
  - Accept user request to join
  - Decline user request to join
  - Delete user from org
  - Request to join University event
  - Specify how many people or spots in request
  - Edit request to join university event
  - Change the amount of people or spots in request
  - Delete request to join university event
- Delete Account

University

- Register
  - Select university
- Login
  - JWT (java web token) given to users based on organization and university.
- University functionality
  - Decline org request to join
  - Accept org request to join
  - Delete org from university listing
  - Delete user from org
  - Create university event
  - Specify which land will be used
  - Edit University event
  - Change the amount time or place for an event
  - Delete request to join university event
  - Certain organizations might get removed by university for specific events
- Delete Account

7.3 API Specifications

We will use a Google API that provides the coordinates of a shape on the Google map. As of right now we will not provide any APIs for third parties to use. All of the processes will happen through the GUI and interactions with the web server.

7.4 Technologies and Implementation

University admins, Organization admins, and users will each have a separate GUI to interact with. The front end will be implemented with React with dynamic content coming in the form of a JSON from the server. Our application will make use of HTTP requests to interact with the web server. Users and admins will have login credentials stored in the database and authentication tokens will be generated based on their role which will give them access to certain routes and resources. We will make use of JWT to generate those tokens. Users will make use of CRUD operations to modify resources. We will make use of Flask web framework and its built in functionalities such as routing, object relational mapping, form processing and session management. Our web server will interact with a google API that will send back coordinates of a shape on google maps in JSON format. Any browser that can run JavaScript will be able to connect to our web application. We will use Sqlite as our database engine. In the development stage we will use the databases locally and in deployment we will use AWS RDS to host our database. We will use AWS beanstalk to

host our web server for users. The servers that will host our application will be managed by AWS. No specific hardware except a computer that can connect to WIFI and has a browser is required.

7.5 Data Design

Since our data has a structured format, we have decided to use a relational database for our data. Among the relational database engines, Sqlite was the best choice as it is efficient for projects of this scale. We will use AWS RDS as it is fully managed and makes the management of the database easier and we won't have to worry about the infrastructure and patching of the database.
Our database will contain the following main models: University, Organization, User (Student), Roles, Events, Plots, and Reservation.

- /API
- /users POST
  - /register
  - /select-university
  - /login
  - /jwt
- /org-functionality POST
  - /request-to-join

8 EVALUATION

The goal of evaluation is to ensure that our problem statements are being fulfilled through functionality testing and are accessible to the user through usability testing. This includes usability for people with disabilities as well as the normal user with layout and good function feedback.

8.1 Functionality Evaluation

The overarching goal of functionality testing was to accelerate the development process as well as make sure the algorithm works and all user functionality is capable while being error-free. Different forms of testing on different systems were used to allow this to happen but the main goal was to make a usable system for the end user most optimally for the developers.

8.1.1 Evaluation Procedures.

Unit testing: We have three main systems being built at once during the process of this project one is the frontend, one is backend/api development, and lastly is the algorithm. Each one of these systems is being tested uniquely for that given system's technical abilities. The front end had more unit testing based on functionality and mobility between pages. There was also static data implemented into the code to test certain functionality before API implementation. This allowed for a more accelerated software development process that didn't have as many bottlenecks. For the API we used Postman to test each API call before connecting it to the front end. For the algorithm, we created certain test cases that tested the algorithm directly without any implementation to make sure it was giving the idea output. We created a lot of unit test cases for the algorithm to make sure it worked and tried to cover as many edge cases as we could think of.
Integration testing: After we had certain components that go together complemented we implemented them together and tested them through the front end interface. We thought of unit test cases like before but used the front end to input the data. This allowed for more testing of the functions while also testing that the components were working properly together and helped us work towards a usable product.
System testing: After we have a fully integrated system we conduct testing through using the normal functionality of the system and making sure that everything works as it's supposed to. Our team would interact with the system and try to find ways to break it through interactions we did not think of. Also, we had outside people interact with it to find ways that we did not think of to interact with the system.

8.1.2 Evaluation Results and Discussion. Usability: Our user tests were a success, with 100% of university testers, 80% of organization testers, and 100% of student testers being satisfied with the result of the system. This means that our system provides users with a

platform to perform the tasks in seconds rather than days, and makes their lives easier, which is what we originally set out to do. Along with this, 90% of users deemed our algorithm's output to be fair. This was of vital importance to our project. One of the driving forces of our approach was to make the existing method more fair for all parties involved, which meant allocation of land dependant on average member response time and number of members attending any given event. Some qualitative feedback we received was that "it was a fair system" and "I like the clear language", making it "easy to use". This proved that we accomplished what our original definition of success was and that we served our users appropriately.
Functionality: For the functionality testing of our system, we implemented a full testing suite comprised of unit tests and integration tests. These ensure that our system functions the way we designed it to and delivers consistent, accurate results to our users through a seamless, continuous experience. The testing suite covers our API routes related to registration of our 3 types of users, as well as any actions taken by the users on our site. The algorithm is tested given mock data to test performance and the results are positive, with all aspects of our project working seamlessly. With an average runtime of about 0.003 seconds per person attending the event, our algorithm and system decreases the time taken for an individual to receive a plot of land from days to seconds.

8.2 Usability Evaluation

Our usability evaluation will involve 20 individuals.

8.2.1 Evaluation Procedures. User study target group: The 20 chosen individuals will be selected accordingly, with 10 individual students, 8 organization officers, and 2 university officials. We will get members of different organizations to evaluate the needs of varying groups.
User study recruitment: The participants will be found via our group members and some organizations they are a part of, as well as through Zachry 450, which is where multiple organization's offices are located. Various emails will be sent out to ensure we gather sufficient participants that cover all aspects of the expected user base.
User study protocol: The study will be performed in study rooms found within buildings on campus. The participants will use a laptop provided with the interface and they will be asked to complete tasks according to their role, either student, organization officer, or university official. We will have the participant talk out loud as they complete the task to better understand the user journey. There will be a post interview to gather more information about the user experience and overall feedback. The process will take anywhere from 5-10 minutes.
Planned Questions: What did you think about the overall process? How does this process function in comparison to the existing one, how is it better or worse? Which aspect of the interface did you find the most helpful? Is there something you wish to be added? How often do members of your organization go through the existing process in a semester? (Never, 1 Time, 2-3 Times, 5+ times) How many of those times do you think would have been improved with this new process?(Never, 1 Time, 2-3 Times, 5+ times)

8.2.2 Evaluation Results and Discussion.

9 DISCUSSION

...

10 FUTURE WORK

...

11 CONCLUSION

...

A PROJECT MANAGEMENT

A.1 Team Agreement

This project signifies our commitment to working as a cohesive unit, tackling challenges head-on and achieving our shared goals. We begin by fostering mutual understanding through individual and collaborative assessments, leveraging Clifton Strengths

analysis to identify individual strengths and how they contribute to the team's success. Open and transparent communication will be paramount, ensuring we build each other up and celebrate individual contributions, fostering a positive and supportive environment.

Knowing our responsibilities is only half the battle. We pledge to hold each other accountable and ensure individual contributions align with the team's objectives. The project manager and progress tracking tools will serve as our accountability framework, while open communication will facilitate timely requests for assistance.

Our aspirations are ambitious: building a fully functional, seamless full-stack application. This translates to enhancing the user experience for student organizations and ultimately enriching the university's cherished tailgating tradition. We dedicate class time throughout the semester to this project, with additional Friday meetings at 3 PM scheduled if necessary.

By signing this agreement, we acknowledge and accept the outlined terms, pledging our best efforts to uphold them and ensure a successful, collaborative journey towards achieving our shared goals.

Chandler Stanchfield
Gian Rodriguez
Justin Kline
Sasan Lotfi
Wyatt Griffin

A.2 Roles and Responsibilities

Justin Kline: *Manager and Back-end Developer*

Project Leadership: Justin will act as the project manager, spearheading the overall planning, execution, and delivery of the project. He will be responsible for creating comprehensive project plans, defining tasks with clear objectives and deliverables, and ensuring the project adheres to both schedule and budget constraints.

Stakeholder Engagement: As the primary point of contact for stakeholders, Justin will manage their expectations, actively gather requirements, and provide regular updates on progress, ensuring transparency and fostering collaboration.

Communication Facilitation: Recognizing the importance of seamless information flow, Justin will facilitate effective communication channels among team members. He will actively listen, address concerns, and proactively resolve any conflicts that may arise, promoting a collaborative and productive team environment.

Detailed Documentation: To ensure project clarity and accountability, Justin will manage all project documentation. This includes maintaining requirements documents, meeting minutes, and comprehensive progress reports that capture key decisions, milestones, and action items.

Quality Assurance Champion: Committed to delivering high-quality work, Justin will champion and oversee established quality assurance processes. He will verify that deliverables meet agreed-upon standards and client expectations, ensuring a final product that exceeds expectations.

Proactive Schedule Management: Understanding the importance of timeliness, Justin will closely monitor project timelines, proactively identify potential delays, and implement necessary adjustments to keep the project on track and within the set time frame.

Scope Management Expertise: Justin will meticulously manage the project scope, ensuring that all deliverables align with the established objectives and requirements. He will actively monitor project boundaries and make informed decisions to maintain focus and avoid scope creep.

Risk Mitigation Champion: Recognizing the inherent risks in project management, Justin will proactively identify and assess potential risks. He will develop and implement effective mitigation strategies, continuously monitor risk throughout the project life cycle, and take action to minimize potential impact.

Gian Rodriguez: *Full-stack Developer and Algorithm Lead*

Development Dynamo: Gian will spearhead the project's technical development, designing, developing, and rigorously testing both the user-facing front-end and the robust back-end components of the application. His expertise on clean, efficient code will serve as the core of the algorithm that partitions sections.

Collaborative Spirit: Recognizing the importance of a unified approach, Gian will actively collaborate with other team members throughout the development process. They will foster seamless integration of front-end and back-end components, ensuring alignment with project requirements and overall success. This collaborative spirit will guarantee a cohesive application that meets all stakeholders' needs and expectations.

Wyatt Griffin: *Full-stack Developer and API lead*

Development Dynamo: Wyatt will spearhead the project's technical development, designing, developing, and rigorously testing both the user-facing front-end and the robust back-end components of the application. His expertise in various programming languages, frameworks, and technologies ensures efficient and high-quality code, laying the foundation for a seamless and reliable user experience.

Collaborative Spirit: Recognizing the importance of a unified approach, Wyatt will actively collaborate with other team members throughout the development process. They will foster seamless integration of front-end and back-end components, ensuring alignment with project requirements and overall success. This collaborative spirit will guarantee a cohesive application that meets all stakeholders' needs and expectations.

Chandler Stanchfield: *Front-end Developer and UI/UX Designer*

Front-End Architect: Chandler will take the lead in crafting the user experience, focusing on developing user-facing features that are both functional and intuitive. His expertise will ensure a seamless interaction flow that minimizes friction and maximizes user satisfaction.

UI/UX Mastermind: As the UI/UX designer, Chandler will be responsible for the visual identity of the application. He will create wire frames, prototypes, and visual designs that not only appeal to the eye but also align perfectly with user needs and business requirements. This ensures a user interface that is both aesthetically pleasing and functionally efficient.

Collaborative Spirit: Recognizing the importance of a unified user experience, Chandler will actively collaborate with back-end developers. This joint effort will guarantee seamless integration of front-end components with back-end systems, ensuring a cohesive and intuitive journey for all users.

Sasan Lotfi: *Back-end Developer*

Back-End Maestro: Sasan will serve as the architect of the application's core functionality, focusing on developing robust server-side logic, databases, and APIs. His expertise will ensure the application operates seamlessly and efficiently, delivering a reliable experience for all users.

Database Strategist: As the database management lead, Sasan will design and optimize the application's database schema. This involves ensuring data integrity, maintaining optimal performance, and facilitating scalability to accommodate future growth. His meticulous approach will guarantee the smooth operation and efficient handling of all information within the application.

Collaborative Bridge: Recognizing the importance of a unified system, Sasan will closely collaborate with front-end developers. This joint effort will focus on seamlessly integrating back-end systems with front-end components, ensuring smooth data flow and functionality across the entire application. This collaborative spirit will ensure a cohesive user experience that delivers on its promise.

A.3 Software Development Methodology

Through collaborative discussions, our team recognized that an Agile approach aligns more effectively with the dynamic nature of our project compared to a Waterfall methodology. While the Agile manifesto provides a valuable framework, we acknowledge the need to adapt its principles to fit the unique context of our project and academic setting. We will work in 3 sprints, followed by user studies and then a final mini-sprint polishing up the work done. After each sprint, we will check each aspect of our project, and revisit the drawing board as needed, making changes along the way to ensure the UI/UX and the algorithm are up to the standard we have set.

- **Satisfy the customer through early and continuous delivery of valuable software**
  While we don't have a traditional customer, we can focus on delivering value to the instructor and the learning objectives by

demonstrating progress through frequent updates and presentations. We can also gather feedback from student organizations to incorporate their needs and improve the final product.

- **Welcome changing requirements, even late in development**
  Recognizing our limited knowledge and evolving vision, we welcome changes to requirements and stories throughout the development process. This flexibility allows us to adapt to new information and feedback.
- **Deliver working software frequently, with a preference for short timescales**
  Dividing the project into smaller, manageable milestones will showcase our progress and allow for early feedback from the instructor and potential users. This iterative approach ensures we're on the right track before moving onto more complex features.
- **Collaboration between business people and developers throughout the project**
  While we lack direct interaction with organizations, we can leverage existing relationships to gather ongoing feedback and suggestions for improvement. Additionally, we can maintain open communication with the instructor to ensure alignment with requirements.
- **Self-organizing teams make the best architectures, designs, and decisions**
  Working as a self-organizing team empowers us to make efficient decisions about design, architecture, and task allocation. Regular communication and established collaboration practices are crucial for maintaining a productive and efficient team environment.
- **Maintain a sustainable pace of work for the team**
  Balancing academic demands and project work is essential. We'll set realistic goals and deadlines and establish a sustainable pace of work that avoids burnout while prioritizing learning and collaboration. By adapting these Agile principles to our specific context, we can ensure a successful and rewarding project experience for both ourselves and the stakeholders involved.

A.4 Implementation Schedule

Our schedule is split into 3 sprints and 2 final phases:

Weeks 4-6: Sprint 1
This sprint focused on setting the foundation of the project, from the database to the tools and technologies that we will use throughout. Originally, we believed we would use flask to construct the app, but then pivoted to django because of its built in functionality and improved database connectivity. This is where we settled on using python for our algorithm and leveraging the Google Maps API.

Weeks 7-9: Sprint 2
This sprint focused on implementation of the project. We worked on creating an MVP of our front end and a separate one of our backend. Our approach to the backend here was naive and iterative, and our front end was still very barebones, but we had two working independent components.

Weeks 10-12: Sprint 3
This final sprint focused on integration and improvements. We combined the two facets of the project into one functioning result and knew there was more to be done. We went back to the drawing board and thought of ideas to make our algorithm more efficient and fair. This was done by creating a queue based on average member response time and then running a BFS on a grid we create, generating the plots.

Weeks 13: Project Evaluation
This phase focused on the evaluation of our project, both functional and usability-focused. We performed user studies on 25 people, testing all 3 types of users that will use our application. We also tested the fairness of our algorithm's output. Our functionality tests were created to ensure that our software worked as desired, and our coverage did exactly that.

Weeks 14: Project Finalization
This final phase of the semester is where we polished up our product, final presentation, and the report you're reading now. It was about showcasing our efforts done in the right light and making sure all 5 of us were more than satisfied with the output.

A.5 Budget

Internal course resource needs: For the purposes of the project within this course, we need access to certain software, including django for constructing the application and python for algorithmic purposes. Along with this, the Google Maps API will be leveraged for section selection. All of these technologies are free and easily accessible. We will be hosting the project locally, but in terms of it expanding and being used in a larger scale, we would look into cloud services such as AWS. Cost for an outside company: If an outside company were to tackle this project, the costs would be spread out in several areas. These include the hosting of the website and the database, along with salaries of any employees hired for the project. Developmental costs include salary of 3 or more software engineers, and depending on the size of the company, the hourly wage of these employees could vary from $25 - $60 an hour. With an expected person-hour total of more than 600 hours, this would be $15,00 - $36,000 total for salaries. Laptops for each engineer will be needed as well, with a macbook costing around $2000. The maintenance of hosting a site and database has varying costs as well, with one example being AWS CloudFront, which costs less than a dollar per 1,000 requests.

ITEM	DESCRIPTION	QUANTITY	STOCK NUMBER	UNIT PRICE	TOTAL PRICE	INTENDED PURPOSE	ACTUAL USAGE
SWE	Software engineer creating the project	3+	N/A	$25 to $60 an hour	$75 to $180 an hour	Develop the project	Develop the project
Laptop	Tool to complete the project and run code (macbook)	3+	A2992	$2,000	$6,000+	Develop and run code and connect to company	Develop and run code and connect to company
AWS	Cloud hosting service	1 instance of it for the project	N/A	Depends on usage (CloudFront: <$1 per 1k requests)	See previous cell	To host the website and database	Host the website and database

Fig. 10. Budget Table

A.6 Project Management Artifacts

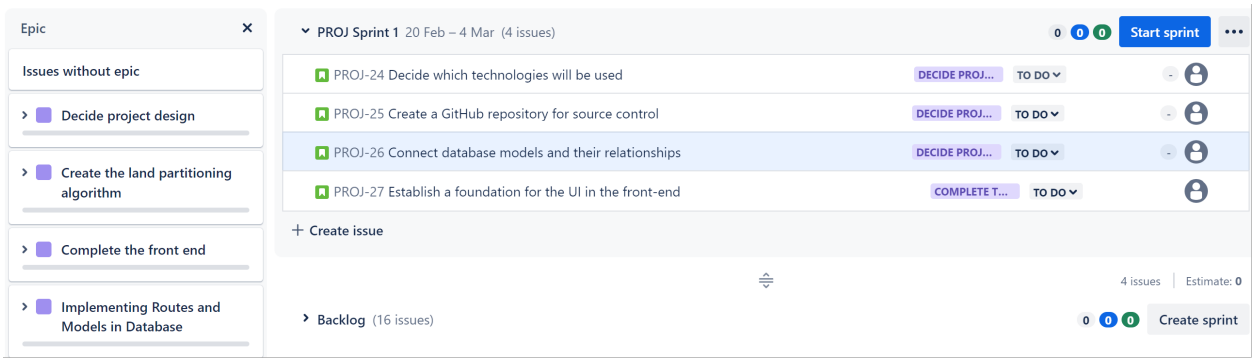

Fig. 11. Jira Project Management

Fig. 12. Gantt Chart

A.7 Software Development Artifacts

Our software development artifacts will consist of a full-stack application and all of its components. These are a functioning back-end that connects seamlessly to a front-end.

B IMPLEMENTATION DETAILS

B.1 Application Layer

B.1.1 Backend. For implementing bookend routes and logic we are making use of Django and Django Rest Framework. Django Rest Framework provides us with built-in features to create rest APIs and return back model data in form of JSON. Database schema and interaction with the database are done through Object Relational Mapping provided by Django framework. The routes allow GET, POST, and PUT requests based on their functionality. We are making use of class based views for the routes and use Django's Token Authentication feature to authenticate users into our system. The sessions are token based and the user should send Token data to get authenticated. For this class we decided to run the application locally and not host it online. However, for

large users it is recommended to host the application on cloud services such as aws that provide scalability and high availability. Since our application will have sudden high traffics during event registration times, it is crucial to have a system that is highly available and accounts for the sudden spark in user traffic.

B.1.2 Partition Algorithm. The partitioning Algorithm will be run in python in the backend and the result will be sent as a picture to the frontend. The algorithm will need average registration time of organizations attending an event, the number of members per organization, and even plot's coordinates to be able to partition the plot associated with the event and create a visual depiction of the partitioned land. The required data will come from database using ORM queries. The area needed for each organization will be determined by that organization's member count and the average registration time. For example, an organization with a lower average registration time of its members will have higher priority in the queue to get a land. Given the coordinates of the edges of the land, the algorithm will create a grid that covers all of the area of the land. This grid consists of squares of same area. The algorithm will run breadth first search on the squares of the grid and will create the partitions as it goes and visits all the squares. As an example, if the first organization in the queue needs 1000 square feet, the algorithm will keep visiting squares and add their area until it reaches 1000. It will mark those squares as organization 1 and will start again to find the next section of land for the next organization.

B.2 Presentation Layer

Frontend user interfaces are implemented using React and JavaScript. React facilitates development of frontend components dynamically. Axios is used to send http requests to backend API and fetch data to be shown to the user. Requests will make use of cookies to send Token information with their requests to get authenticated.

B.3 Data Layer

The data is hosted locally using in-memory sqlite database. Queries are made using Object Relational Mapping. Django allows defining relationships between models using Foreign Keys. Routes are implemented for the users to be able to view and modify the data layer.

B.4 Tools

A working computer to run as a server that would process and store application data. User would use a browser on a computer to access the website and login to the system. The hosting server should have npm, JavaScript, React, Python, Django, Django Rest Framework and Axios installed. GitHub was used as a version control platform for the project.

B.5 Project Development Phase

B.5.1 Sprint 1 (02/20 - 03/04):

backend:

- routes implemented
- database models and relationships completed
- accounts sign up, login, and logout functionality completed

frontend:

- UI for universities, organizations, and users to create, login and logout of their accounts
- UI for designating tailgate land on the map and sending the coordinates to backend

B.5.2 Sprint 2 (03/04 - 03/18):

backend:

- Event creation and notification completed
- Assigning lands to organizations and sending the data back to frontend

- Organizations should be able to register for events

frontend:
- Organizations and universities should see their assigned lands
- Users should be able to register for an event

B.5.3 Sprint 3 (03/18 - 04/01):

backend:
- logic for student registration and average registration time for each organization implemented - continue working on partition algorithm. - Basic partition algorithm that uses incremental approach should be completed for minimum viable product frontend:
- Organizations should be able to register for an event
- Organizations should be able to register for an event through the organization - Universities should be able to close the registration and get the final results of the land

B.6 Project Evaluation

B.6.1 04/02 - 04/16

. - Test system's functionality and analyze the partitioning results based on predefined metrics.
- User studies and data analysis

B.7 Project Finalization and Presentation

B.7.1 04/16 - 04/31

. - Land partitioning algorithm completed
- real coordinates of the lands and designated areas should be sent to the user
- Final report and preparing all final deliverables - Final presentations

C USER'S MANUAL

C.1 Installation Instructions

C.1.1 Site Operators. Site operators should download the application via (placeholder for link).

- **Requirements** Site operators need to ensure the following software versions are installed:
  - **Python Version 3.1X:** Download and install Python from https://www.python.org/downloads/. Verify with the command "python –version"
  - **Pip:** Download and install the latest version of pip. Verify with the command "pip –version"
  - **Node.js Version 20.X:** Download and install Node.js from https://nodejs.org/en/download/. Verify with the command "node –version"
  - **Additional Package Installation** In the root directory, run the run.bat file on Windows 10+ machines. If this fails or you are on another OS, in the root directory, run the command "pip install -r requirements.txt" via a terminal.

C.2 Operating Instructions

- **Site Operators**
  - **Windows Quick-Start:** To run the application on Windows 10+ systems, launch the run.bat file from the root directory. This will automatically start both the front and back-end via the command prompt interface as well as install all necessary modules before launching the web application locally.
  - **React Front-End Setup:** To run the React front-end, the user must follow these steps:
  (1) Navigate to the `static/land-rush` directory.
  (2) Run the following commands:
    * `npm install`
    * `npm start`
  - **Django Back-End Setup:** To run the Django backend, the user must follow these steps:
  (1) Navigate to the `landrush` directory.
  (2) Run the following command:
    * `python manage.py runserver`
  - **Django Database Management:** To sign in to the database, the user must follow these steps:
  (1) launch http://127.0.0.1:8000/admin in a browser while the back-end is running.
  (2) use a valid set of credentials to access the database where information can be modified manually as needed.
- **End Users (Work In Progress as UI appearance Updates)**
  - **Access the website:** Follow the link here (no currently hosted site): localhost:3000

(1) **Getting Started**
  - **Students and Organization Leaders:** Register as a student via the Register Student link in the top navigation bar.
  - **Universities:** Register as a new college or university via the Register University link in the top navigation bar.
  - **Existing Users:** Register as a new college or university via the Register University link in the top navigation bar.

(2) **Students**
  - Register for and view upcoming events with your organizations.
    * Click register or unregister on an event to allow students to register for an event with your organization.
      · Additional Notes:
      · You can register for events with an organization of your choosing 7 days before an event up until 2 days prior to the event.

(3) **Organization Admins**
  - Register for and view upcoming events with your organizations.
    * Click register/unregister on an event to allow students to register for an event with your organization.
  - Additional Notes:
    * You can register for events with an organization of your choosing 7 days before an event up until 2 days prior to the event.

(4) **University Users**
  (a) **Land Plots**
    (i) **Create and Modify Plots to use for Events.**
      – Select a Plot
        * Select "Create New Plot" to begin creating a new plot
        * Alternatively, select one of the pre-existing plots from the list shown beneath the search bar to modify or view it.
      – Adjusting Coordinate Points
        * Click draw/modify points depending on whether you're creating or updating a plot.
        * Left-click on the map to add a point connected to the previously placed point.
        * Left-click on a transparent middle-point between two points to insert a new one between them.
        * Right-click on the map to remove the last point placed from the polygon.
        * Move points by dragging them.
        * Move the entire plot by dragging any part of it within the highlighted plot.
      – Create or Update a Plot
        * Clicking Create/Update Plot will finalize the coordinates on the map and update the database.
        * Clicking Delete Plot will erase it from the database.
  (b) **Events**
    (i) **Designate Event Details for your Students.**
      – Create or Modify an Event
        * Select "Create New Event" to begin creating a new event
        * Select one of the pre-existing events from the list shown beneath the search bar to view it.
        * Clicking Delete Event will erase it from the database.
      – Running the plot algorithm manually.
        * Our algorithm automatically fills the plot once registration ends.
        * Select Fill Plot to run the algorithm manually with currently registered users.
        * Select View Plot once the algorithm has completed to observe the results.
  (c) **Organizations**
    (i) **View and Remove Organizations within your University.**
      – Kick an Organization
        * Clicking Delete will erase the associated organization from the database.

ACKNOWLEDGMENTS

...

REFERENCES