Team Name: The Rolling Crew
Team Members: ~~Tanishka Mehta, Shreya Devaraju, Alvin Le~~

Tanishka Mehta
903878002
tmehta46@gatech.edu

Shreya Devaraju
9033753758
sdevaraju6@gatech.edu

Alvin Le
678 733 4826
ah307@gatech.edu

App Name: RollCall

Team Agreement

Team Goals: Goals for this team include completing all project sprint deliverables on time, contributing equally to tasks, maintaining clear communication, meeting consistently throughout the sprint, making decisions as a team, learning new concepts and tools, and gaining meaningful hands-on experience.

Participation: Each team member will contribute equally to tasks, discussions, and deliverables. For team sprint, an internal, flexible deadline will be created for tasks. Team members are expected to complete assigned work on time and communicate if challenges or setbacks arise.

Communication: Primary communication will be through an iMessage chat. To assign tasks, Google Docs' comment tagging feature will be used. For communication with the group and Teaching Team, Microsoft Teams will be used. If someone has to complete a section needed to start another section, an iMessage will be sent in the group chat. In case of this, the person must respond within 6 hours and have their part completed within 12 hours. When code is pushed or merged into *main* (on Github), a text must be sent within the hour informing the remaining team members to pull the most recent code. This is to help avoid complex rebases later down the road.

Meetings: Team members are expected to attend all meetings and complete work in between meetings. There will be one meeting scheduled at the beginning of the Sprint to go over expectations and deliverables, as well as internal deadlines. There will be one meeting held the Saturday before the Sprint is complete in order to ensure all deliverables are complete. ~~Weekly meetings will be held Mondays and Wednesdays on Google Meet from 9 PM - 10:30 PM.~~ Additional meetings may be scheduled as deliverable deadlines approach. These meetings will be scheduled collaboratively to accommodate each team member's schedule. In case a team member cannot attend a meeting, they must inform the team at least 6 hours in advance and another time for the meeting will be decided.

Decision-Making: All decisions will be made with a unanimous vote. Critical decisions will be documented in a shared notes document for transparency and record-keeping. Decisions will be made by attempting to integrate all team members' opinions into the final product, as much as possible. However, the main priority will remain achieving a unanimous decision.

Tools: Tools we will use include a GitHub repository for all code and documentation. We will also use Google Drive to collaborate on team deliverables, such as the notebooks and presentation slides. Team members must also refer to Canvas Discussions for feedback, as well as Microsoft Teams to share Sprint materials with the rest of the class.

Conflict Resolution: First, team members will foster open communication amongst ourselves to resolve disputes. These conversations will be open, professional, and respectful, with the third group member not involved in the conflict acting as an unbiased mediator. If this method does not work, the issue will be escalated to the CS 4261 Teaching Team.

Agreement Updates: As the project evolves, this agreement will grow with us. It will be revisited at the beginning of each sprint, and any changes (highlighted in yellow each sprint) will require agreement from all team members.

Signed by: Tanishka Mehta, Shreya Devaraju, Alvin Le
Signed Date (Sprint 3 Updates): October 9, 2025

No updates were made to the team agreement in Sprint 5.

Problem Overview

Food trucks are a quickly growing industry, having increased in revenue by 300% in the past 3 years (uschamberfoundation, 2025) and are projected to grow to to $6.5 billion globally by 2027 (Allied Market Research, 2023) . Food trucks are a major part of urban food culture, especially in large cities where people seek quick, affordable, and diverse dining options, but customers still lack reliable, real time ways to find trucks. Foodies and people looking for unique places to eat struggle to find real time details on where food trucks, pop-ups, and stalls are currently, while vendors struggle to reach nearby customers during the limited amount of time that they're in the area. Unlike restaurants with fixed addresses, trucks are mobile and often change spots throughout the day based on permits, foot traffic, and events. Customers may find that a truck is already closed for the day or has changed locations by the time that they get there. Existing channels (instagram stories/posts, TikToks, and city website schedule listings) are often incomplete, not up to date, or fail to reach their intended audiences in time, leading to missed sales for vendors and missed meals for customers.

Due to day to day uncertainties, Food truck vendors often struggle with managing inventory management, either by over stocking ingredients that lead to food waste or under stocking and missing sales profits, due vendors lacking the data analytics to understand customer flow patterns, peak demand times, or geographic performance. The industry currently doesn't have a definitive mobile solution and with 85% of Americans owning smartphones (Pew Research Center, 2021), a mobile solution that helps with real time decision making capabilities would greatly help both vendors and customers. These discovery limitations means that even in dense urban areas, vendors remain invisible to potential customers within a few blocks, creating artificial barriers in the mobile business model.

Domain Research

Current solutions attempt to solve this issue, but have significant shortcomings. For example, many food trucks that post their locations on social media platforms tend to have inconsistent updates. City websites also publish food truck schedules, but they rarely provide live updates. General food discovery apps, like Yelp and Google Maps, list food trucks as businesses and only display a singular static location, which fails to capture the dynamic nature of a food truck's movement. Although there are some apps and websites that are dedicated for food trucks, they are not universally adopted by vendors or customers. This creates a fragmented experience where customers might need to check multiple platforms to locate food trucks.

Even though food trucks serve 11 million people a day, on average, there is no widely-adopted, real-time solution that allows customers to easily track where food trucks are operating at any given time ("*2024 statistics"*, 2024). A successful solution must overcome vendor adoption barriers, provide accurate, real-time location tracking, and present it to customers in a simple way.

Food truck operations are also shaped by culture and consumer behavior over time. In a fast-paced world, customers increasingly expect real-time information for everyday decisions, from ridesharing to food deliveries. This extends to food vendors as well. Many food trucks rely on foot traffic and word of mouth for business growth; moreover, people prefer food trucks since they are a fast and affordable option. This creates frustration when customers cannot access accurate, real-time updates. Although there are many live-tracking services that exist, this solution has not yet been widely applied in the food truck space. This gap between consumer expectations and vendor adoption highlights why the problem persists and creates a perfect opportunity for innovation in this area.

Despite customer demand increasing, convincing vendors to join new platforms still remains a major roadblock. Many food truck vendors independently run their trucks and have limited time or technical resources. Juggling cooking, logistics, and daily operations leaves little time to manage a digital platform. Many vendors are skeptical about whether adopting a new system will actually bring more customers or simply add extra work with little reward. Without clear incentives, an easy onboarding process, or evidence of customer reach, vendors hesitate to participate, which in turn limits the effects needed for a food truck app to succeed.

As we continued to develop the app, and compare with other apps that we use daily, we noticed a trend. We needed a lightweight app. We prioritized simplicity and basic functionality over fluff and features that have little value.

Most apps also generate revenue through ads. Since this app does not have any internal cost to users, we explored looking at ways to host ads on our app.

Lastly, to enhance the app in the future, there need to be metrics. This allows developers to see how many people use the app, which features are useful, what trucks are best, etc. Having these insights can steer application and feature enhancements.

Competitive Analysis

To gain a comprehensive picture of the market and our competitors, our team conducted research on the different websites and apps which have made impactful efforts to address this issue, both national and local. In doing so, we hoped to better understand the strengths and weaknesses of our potential competitors and better determine what we could offer that these solutions don't. The main competitors we analyzed include Roaming Hunger, a food truck booking service; Street Food Finder, a food truck tracking service; Marketplace Events, a dedicated events booking page on Facebook; and Access Atlanta, a local guide for events and food pop-ups in Atlanta. ~~In Sprint 2, based on class feedback, we additionally studied social media as a general competitor group since consumers often first engage with food trucks through marketing efforts on social media. This allowed us to gain a more comprehensive understanding of what specific information vendors consider important to reach customers, which platforms they use the most, and what regular actions they take on these applications that would be helpful to integrate into our chosen approach.~~

Strengths & Essential Elements of Competitors:

- Event Integration & Tools: Most of these apps (Roaming Hunger, SFF, Facebook Marketplace Events etc.) provide tools that support events, rallies, catering coordination not just for food pop-ups but also for other types of events giving them a larger customer base and boosting activity
- Local/Community Trust: For Marketplace Events, Access Atlanta: having community curated picks, fan promotions, and "influencer" approval can appeal more to users looking for.
- Live Tracking: Street Food Finder and Roaming Hunger in particular thrive on this aspect so that users can track where the food trucks they're looking for are in real time in advance (Ettinger, 2024, para. 4).
- Marketing, Advertising & Outreach: From the vendor perspective, applications like Instagram, Facebook, and Tiktok gives vendors the ability to showcase their food, give updates on location or events, and create content (short form or long form) that connects them with more users.

Weaknesses of Competitors:

- Lack of Coverage & Reliability: Lots of food trucks don't sign up for platforms like these (due to preference for a different fit) or update regularly leading to stale listings and user frustration because they don't see a lot of options (i.e. the food truck they love might not be listed/active on one app/platform so they need to go to another). Vendors also express frustration with having to post updates on multiple applications
- Social Media Competition: Lots of apps who follow this type of model (Roaming Hunger, Street Food Finder) are now competing with a thriving social media space which most users rely on to find local pop-ups/food trucks in their area or keep track of events (marketplace events, access atl instagram account). Because social media offers more curated content, vendors can showcase their unique offering and make more users who share content with each other motivated to try food trucks together.
- Geographic Skew for Big Cities: These sorts of apps do really well in bigger cities but not so much in rural areas where food trucks aren't active so without enough vendors/food trucks in an area the app won't gain traction there, giving these apps a big limitation.

Ultimately, none of these applications are comprehensive so most people (both consumers and vendors) end up using some combination of them or even just relying on word of mouth or spur of the moment decisions based on where they are.

Summary:

- Roaming Hunger: National Directory (large coverage in the US)
- SFF: Live Tracking & Locations

- Access Atlanta: Local Curation And Reviews
- Marketplace Events: Community Driven
- ~~Social Media (Instagram, Tiktok): Visual Content, Increased Brand Awareness, & Targeted Outreach~~

Customer Discovery Interviews

Interview 1: A student at Georgia Tech said they only eat at a food truck if it happens to be along their route. Since there aren't many apps in Atlanta to track food trucks, it feels like more work to find a truck compared to just ordering takeout. We learned that convenience and low search effort are critical factors in whether people choose food trucks.

Interview 2: A food truck owner said that they often forget to update the truck's Instagram because they are usually very busy. They said they would rather sacrifice a few potential customers that rely on updates to serve other customers than post live updates during busy hours. We learned that vendors prioritize in-person service over digital, potential customers. This creates reliability gaps for customers trying to track their favorite trucks.

Interview 3: A full-time working professional shared that when grabbing lunch with co-workers, they prefer quick options like food trucks. However, since there isn't a central app for food trucks, it can be frustrating and complicated to choose one to go to and end up going to fast-casual restaurants instead. This highlights the need for a universal app that has accurate location tracking in order to improve customer experience.

Interview 4: A vendor said that they usually have a limited amount of space for supplies that they carry on them, and will leave once they have run out. On the busy days in the week they typically can go through all their supplies, while on less busy ones they'll leave after a set time. Vendors that are popular and sell out seem to not have much of a need to advertise/promote their pop ups, instead relying on word of mouth. While vendors that are struggling to get enough customers might have a greater need to promote themselves or make it easier for customers to find them.

Interview 5: Another food truck owner wants to gain more visibility and values finding ways to organically reach customers. They like social media apps specifically because it feels like "word of mouth" to them and because it's free (doesn't require a cost for listing). However, social media isn't a good way for them to be discovered in real time. They would like to be able to manage word of mouth and real-time concurrently, but more than any of this they hate having to manage multiple apps and the time drain this adds to their regular operations.

Interview 6: Another Georgia Tech student loves grabbing a bite to eat at food trucks in between classes. Their favorites are "Bento Bus" and "6Pack Vietnamese Food Trucks" but these trucks don't often appear in the exact same spot during the week and they're usually not able to keep up with their schedule (not always listed on a website): sometimes they'll try to take a picture but usually it's easy to lose track of. They also complain about prices at food trucks usually being

high as they need to budget so they don't often eat at food trucks although they would like to try out more.

Interview 7: An office worker living in the suburbs with her family shared that she doesn't usually eat at food trucks because there aren't a lot in places that aren't bigger cities. She mentioned that when she used to live in the city, there were more food trucks available nearby that were easy to try out on the go during a lunch break. She did mention that now that she's moved out to the suburbs, she wants to find more local events for her and her family to try out. She usually uses the internet (local sites) or word of mouth to find out what's going on in her area.

Interview 8: An office worker who just moved to New York City said he's loved getting to try out all the new food trucks in the area. Usually he's moving on foot to get back to his office after getting a quick lunch from a food truck or he doesn't have time to make dinner so he'll grab something quick on the way back home. He says he's aware of different apps like Roaming Hunger and Street Food Finder but after trying them out for a week he said he stopped using the apps because he didn't need to use them regularly. One important thing he mentioned was that he liked getting to "discover" things more than looking them up: when he did try a food truck, he said getting something quick and new was more important than getting something even great so he's fine trying something without knowing the reviews as long as the wait time is small.

Interview 9: A vendor says that they will usually join markets or events along with other vendors as these events usually will promote themselves and gain a lot of foot traffic. These events often will take extra work to coordinate, include fees, and only happen once a while on special occasions so they are always the best long term strategies. An app that regularly promotes their business and location could potentially be a better long term solution.

Interview 10: A graduate student mentioned that food trucks are one of her favorite options between classes or during study breaks, especially since there are usually options on campus that are pretty fast. She explained that she usually finds trucks by walking past their usual spots, but this can be unreliable since trucks are not always there on a given day and there isn't a centralized place to find their schedules. She said she doesn't want to follow them on Instagram because it clutters her feed, and she often misses posts during the day anyway. She was interested in having a singular app that showed what trucks are nearby. Given that food trucks are appealing when searching for a quick option, she wanted to make sure she didn't have time to wander around looking for options.

Interview 11: A vendor who operates during lunch hours in Midtown Atlanta described that during lunch rush, it was difficult to keep updates consistent. He said that social media works well for marketing photos and promotions, but during peak service he rarely has time to post

location changes without sacrificing the appeal of food trucks, their quickness. He noticed that customers often message him directly asking if he's open, which interrupts his workflow and creates extra work for him to reply to each person; if he doesn't reply, he could miss out on potential customers. He emphasized that having a single system where he could log his location once and automatically sync it to both a centralized app and his social media would save time and reduce confusion. He also noted that this could help him reach new customers who don't already follow him online, something his current social media presence fails to capture.

Interview 12: A food truck vendor in his late 50s shared that he has been running his truck for more than 20 years, significantly before social media became a major marketing tool. He explained that his customer base has mostly grown through word of mouth and repeat visitors at the same locations. Even though he has a Facebook page, he rarely updates it and relies more on having a consistent weekly routine. When we asked about using a new app to manage his locations, he was opposed to the idea and didn't want to learn another piece of technology since it would be frustrating and time-consuming. He mentioned that he is already overwhelmed by keeping up with online orders and basic credit card systems. For him, a new app would feel like an extra burden rather than a solution.

Analysis

A common theme among the interview especially for customers is convenience and low search effort was an important factor when choosing to go to a food truck. Students and office workers would often only buy from trucks with short waits, are on the way or nearby, otherwise they would just go to something fast or casual. Some customers enjoyed discovering new food trucks, while other students cared more about how cheap the prices were. Others would rely on events or word-of-mouth to find new food trucks. Vendors would often prefer to focus on running their shops and providing services during operating hours instead of updating their socials. Popular trucks usually relied on events or word of mouth to gain customers. Vendors preferred not managing multiple apps or paying fees. Also operational obstacles such as limited storage, selling out, schedule changes, and leaving early on slow days, would mean that status updates would need to reflect inventory and timing, not just planned hours.

The interviews done in Sprint 2 reinforces the need for a solution that minimizes extra effort for vendors while still providing customers with timely, accurate updates. It also highlights why centralizing vendor updates while allowing customers to validate and contribute information can address both sides of the problem more effectively than relying on a single source. The last interview (interview 12) gave a perspective that was not considered before; however, it emphasized that the app must remain lightweight and easy to adopt, especially for older vendors who may be hesitant to learn new technologies.

Discussions

In our early discussions, the assignment guidelines felt confusing, so we scheduled a meeting with a TA to clarify expectations, which helped us better understand the scope and deliverables. During brainstorming, we debated between our top two ideas: a food truck app to help track vendors or an investment app aimed at simplifying financial decisions. After weighing both ideas, we agreed the food truck problem felt more tangible and easier to validate with users.

In our Sprint 2 discussions, we had to make a lot of decisions as a team. After looking at our notebook again, we realized that we already had a lot of interviews. Therefore, as a team, we decided to only conduct a few more interviews and instead, focus on getting responses for the survey. We also liked all three approaches we came up with in Sprint 1, and initially wanted to combine all of them into one app. However, we realized that having live location-tracking defeats the point of needing social media syncing, especially since in order to get updates on social media, users would already know what specific food truck they want to dine at. We wanted our solution to be more generalized and help people find any food trucks near them. We also talked about additional features we could include in the app on top of live location-tracking, such as notifications for favorite trucks nearby and newsletter for events with food trucks. These are possible future avenues we may pursue to enhance the app. From class, we learned that we don't just want to achieve our goal, but go beyond that and help make the user's experience more convenient.

No changes were made Sprint 4 (The scope of the project did not need to be refined further this sprint as we just continued coding features and developing the app based on our findings from past sprints.)

Solution Approach

Sprint 4: Enhanced Code Development

Process and Discussion: In Sprint 3, we developed the MVP for our app. We had implemented the map and reporting features, as well as authentication. This sprint focused on enhancing our app to be easier to use and more enhanced in terms of the UI. For example, in Sprint 3, we had only 1 food truck reporting page, which was the same for users and vendors. However, after internal testing, as well as feedback from mock vendor testing, we realized that it was tedious to have vendors input their truck name, cuisine, etc. every time they wanted to report their truck. This led to a discussion about how to make the entire app easier to use for Sprint 4, where we focused on tailoring and enhancement. We decided to personalize each of the pages (map, reporting, and profile) to fit the specific user vs. vendor needs. We also brainstormed ways to create the best user-friendly UI, which we refined using A/B testing.

Feedback Integration: We received great feedback from our classmates and the teaching team. Most of the feedback was about user retention, which would be improved by having a sleek, easy-to-use app. Having pages that are personalized would make users want to continue using the app. Other students pointed out, as we also discussed as a team internally, that vendors having to input the same information every time when reporting is tedious and defeats the purpose of using this app; instead, we decided to implement a profile page. This way, vendors only need to input their location when reporting, while the pins automatically pull their profile information. Adding these enhancements to the MVP from Sprint 3 helped us think ahead from a user's perspective, rather than just from the developer's point of view.

Sprint 3: Initial Code Development

Process and Discussion: In Sprint 2, we tested our approaches and decided to continue with a combination of Approach 1 and Approach 3. Essentially, we wanted to create an app where we could have both vendors and customers put in locations. Throughout this project, from peer feedback, domain research, user interviews, and prototype testing (Sprint 2), we learned that our two major issues for initial app usage were vendor adoption and the "empty map" problem. After discussing our interviews and testing results last sprint, we agreed to create an app that allows crowdsourced information, as well as vendor information. This allows vendors to share location tracking responsibility with customers. This solution will work because if vendors are busy and do not have time to update the app, it is likely that there are a lot of customers there, who may be waiting, and can update the app instead. On the flip side, if a vendor does not have many customers, they can update their location manually since they have more time.

Feedback Integration:
The feedback from class helped us decide what features to implement. We knew we wanted to have crowdsourced information, but students mentioned a verification process for this information for safety purposes. This specifically helped in our design as we decided to implement a threshold for crowdsourced information; for vendors, there would have to be some implementation of verification to prove they run a food truck. A concern we had last sprint was getting vendors to test our app. After mentioning this to our peers, many students suggested testing the vendor functionality ourselves. Since vendors are already very busy, they likely won't have time to thoroughly test our app multiple times. Instead, we decided to have them test it at the end of Sprint 4 or Sprint 5 when our app is almost complete.

Sprint 2: Narrowing it down

Process and Discussion: In Sprint 1, we came up with three different approaches to solve the food truck discovery issue.

- Approach 1: Crowdsourcing location and updates
    - In this approach, we allow customers and vendors to contribute live updates about the locations of food trucks. Based on the interviews, we saw that customers liked the idea of being able to validate information themselves. From the Canvas peer feedback, it seemed that crowdsourcing would make the app feel more engaging. However, classmates also warned about having an empty map while initially rolling out the app. When the app is first launched, there may not be many users to input data, so new users may see very little data.
- Approach 2: Social media platform sync
    - This approach focuses completely on reducing vendor workload by letting them send updates to multiple social media platforms at once. Vendors said they would value not having to spend time creating duplicate posts, and feedback from peers highlighted that this solution has very little competition from existing solutions. However, the drawback is that it only benefits a small group of customers that already follow specific food truck accounts on social platforms. As classmates pointed out, most customers want to discover food trucks nearby, not just track the ones they already know of. This limits the audience and broader appeal of the approach.
- Approach 3: Centralized hub with extra features
    - This approach creates a central platform where vendors can manage their profiles while customers can use the same app for discovery. Domain research showed that this had similar features to Yelp and Google Maps, especially with the favorite locations. However, interviews showed that customers were frustrated with the lack of a comprehensive platform for all food trucks. Peer feedback demonstrated the challenge of adoption by vendors having to manually maintain

profiles and by customers having to install a new app. However, the strength of this approach is that there is a one-stop, dynamic platform.

Feedback Integration:
The feedback from class helped us weigh the feasibility of these approaches against their user value. Multiple peers noted that vendors often skip updates when they are busy, so any solution depending solely on vendors risks consistency. Some peers emphasized having incentives for crowdsourcing, since many users prefer to consume information rather than contribute. For future directions, many peers also suggested looking at event organizers and city officials as stakeholders, both to help seed reliable data and to expand the use cases beyond everyday lunch decisions.

Chosen approach:
Based on these insights, we chose to evolve our approach by combining Approach 1 and Approach 3. This hybrid model creates a centralized platform where vendors can provide updates and customers can discover nearby trucks. It also includes customer contributions to confirm accuracy and share location responsibility between vendors and customers. Social media syncing from Approach 2 is convenient for vendors, but it is still directed towards a very small group of customers who already know which specific food truck they want. Since our interviews showed that most customers are looking for apps to help them discover food trucks instead, we prioritized this approach.

Sprint 1: Brainstorming
Approach 1:
One solution is to develop a mobile app similar to "Waze" that is geared towards food trucks specifically. The app would show the real time location of where a food truck is but with a new key element: crowd sourcing. Users can not only look up where a food truck is and see how long it would take to go there but users can also actively confirm/update a food truck sighting so that the most accurate data is displayed and dynamically updated.

Pros:
- Community Driven: Crowdsourced system allows for true real-time tracking on where a food truck actually is, combatting the problem of stale or unpredictable relocations and needing to rely on vendor updates
    - The app can ask a user to confirm if a food truck is still in the spot where it's currently listed and the user can confirm or deny so the app can adjust in realtime
- Users can look up food trucks that they love and see where they currently are and the time it would take to reach the food truck
- Users can receive suggestions or notifications for which food truck is close to them and where to go based on user history and proximity.

- User Incentives: Like Waze users can get discounts at a food truck (just an idea) or rack up points for confirming/updating an accurate trick citing.

Cons:

- Reliance on User Reported Data: An app like this would rely on user activity so we would need to build a good base of users even at the outset who are motivated to use the app for this sort of solution to gain any kind of traction
- This solution also suffers from a similar problem to Waze, Street Food Finder and other live tracking applications in that there's less incentive for those in rural areas to use an application like this. In an area where not as many users are active, this app would not function optimally.
- Initial Listings would also need to be coordinated with Food Truck Owners (unlike restaurants other geographical locations) there's no API/database to pull from and they would need to find incentive to list.

Difference from Existing Solutions:

Street Food Finder:

- This solution relies primarily on vendor updates rather than user-powered updates for live tracking. Vendors have the option to turn on or off their listing and manually update their location based on where they are during business hours. In contrast, this solution takes advantage of online users and takes the burden off of vendors to continually update their location for users.

Roaming Hunger, Marketplace Events, Access Atlanta:

- Roaming Hunger updates the locations for the food trucks part of their program by using Twitter/social media to approximate where the food truck will be. Also, unlike Marketplace Events, and Access Atlanta this approach is meant to be a realtime, dynamic **mobile app**. It doesn't intend to give any detailed information beyond ratings,timings, and maybe a small promotion the way that these other platforms do (Access Atlanta provides deep dives for instance and Roaming Hunger also doubles as a platform to book catering). Ultimately, these approaches are also more web focused or social media focused which is not the space that this approach intends to fill.

Approach 2:

One solution is to create an app that would allow vendors to enter their location once, and then the app automatically pushes the update to sync to any social media accounts associated with the food truck. This reduces the overhead of posting in multiple places while still serving customers where they already are. The app becomes the central, reliable hub for live food truck tracking, while social media platforms continue serving as outreach channels.

Pros:

- This would reduce the workload for vendors since they no longer have to duplicate posts on multiple platforms.
- This ensures consistency between the app and all social media platforms connected, preventing customers from travelling to outdated locations.
- This allows the vendor to be in full control of what gets shared without requiring users to download a new, specific app for updates.

Cons:

- Customers that don't have any social media or don't follow this specific vendor won't benefit from the updates.
- This still requires vendors to actively enter updates, which may be skipped during peak hours.

Differences from Existing Solutions:

Street Food Finder

- This solution requires adoption from both vendors and customers. The proposed approach is vendor-only, lowering the barrier by letting users continue to use their existing social media presence.

Access Atlanta

- This solution relies on publishers and editors to put out information for food trucks, while the proposed approach allows vendors direct, real-time control of their updates.

Approach 3:

Another solution is to have users and vendors download an app. Upon downloading, vendors will have a profile where they can upload photos, give a description of their food and list their operating hours if they like. They will then share their location with the app, and press when they are operating. Vendors can also add a schedule that will share their location at certain times. The schedule will stay in effect until it is turned off or changed. Customers will also share their location and the app will notify them of the location of any nearby food trucks. Customers can save their favorite food trucks, so that they can get notified when they start operating. Once the vendors are done they can press that they have stopped, and customers will stop being notified.

Pros:

- Real time location of food truck vendors
- List of favorite food trucks
- Profile giving information on the food trucks
- One central app for vendors and customers

Cons:

- Requires vendors to manually update when they are running
- Lots of vendors need download and use for it to be worth it for customers
- Requires customers to download

Differences from Existing Solutions:

Street Food Finder

- This app is similar to the proposed solution in that it requires downloads from both customers and vendors, and for vendors to update their status. The proposed solution differs in that vendor locations are automatically updated in real time so long as they allow it. Vendors can also present their operating hours and make a personalized profile about their business. Customers can save and track their favorite food trucks vendors.

Use Cases

Sprint 5: Minimum Viable Product

Use Case 1: User with admin access using the internal dashboard

- Tanishka is a project coordinator for the food truck platform and signs in using their standard email and password. The app recognizes Tanishka's account as an administrator because their Firestore profile contains the isAdmin flag.
- When Tanishka opens the side menu, she sees an additional entry labeled "Dashboard," which is invisible to regular users.
- She taps it and the dashboard loads, powered by the separate analytics backend. Tanishka reviews total user counts, the number of active vendors, daily reports, and alert patterns.
- Next, Tanishka opens the feature flag panel and toggles an experimental UI meant for improving vendor reporting flow. The change is saved instantly in the admin backend and is read by beta users during app startup.
- Tanishka uses this dashboard weekly to evaluate trends and deploy small A/B tests.

Key features used:

- Admin profile flag
- Protected dashboard route
- Separate analytics backend
- Feature flag controls

Value demonstrated:

- Centralized monitoring
- High-level visibility into user behavior
- Ability to run controlled experiments

Use Case 2: User with admin access using the internal dashboard

- Aisha installs the app to find dessert trucks near her campus. She signs in with her email and completes the standard verification process.
- When Aisha opens the interface, she only sees the features intended for general users: map view, favorites, and reporting tools. (Her Firestore profile contains no admin flag, so the dashboard is automatically hidden.)
- Aisha continues to explore the map, favorite food trucks, and submit basic sightings. The experience is simple and does not expose any analytic or configuration tools.

Key features used:

- Non-admin user profile → Standard user map and reporting features
- Conditional UI navigation

Value demonstrated:

- Keeps sensitive tools secure
- Ensures regular users see only relevant interactions

Use Case 3: Vendor creates an account and uses it as unverified

- Diego owns a smoothie truck and wants to be included in the food truck map. He installs the app and selects the option to register as a vendor.
- During signup, he provides his truck name, cuisine category, and operating hours. Because vendor accounts require approval, the app marks his vendor status as unverified and stores his profile in Firestore.
- When Diego opens the reporting tab, he sees a simplified vendor flow but with limited functionality. His reports are labeled as "vendor pending." He can still explore the app and confirm crowd sightings.

Key features used:

- Vendor account creation
- Pending verification flag
- Limited vendor reporting flow

Value demonstrated:

- Prevents misuse of vendor tools
- Allows vendors to begin onboarding immediately

Use Case 4: Vendor uploads a social media proof image and sees the pending approval page

- Priya runs a taco truck and wants her account verified. After creating her vendor profile, she reaches the step that requests proof of owning a food truck (for security). The app asks her to upload a photo of her truck's Instagram page or another platform showing her business name.
- Priya uploads the screenshot, and the backend stores the image securely. The app then reroutes her to a dedicated "Pending Approval" page.
- This page explains that an admin will review her submission, verify her social media presence, and approve her listing. While waiting, Priya cannot submit public reports or appear on the live map.
- Once the admin team reviews the proof, they flip the verification flag. The next time Priya opens the app, she gains full vendor reporting tools and her truck's pins adopt the vendor-verified icon.

Key features used:

- Vendor document creation
- Social media proof upload

- Pending approval page
- Admin verification flag
- Verified vendor pin styling

Value demonstrated:

- Protects platform credibility
- Prevents fraudulent vendor entries
- Creates structured onboarding for new vendors
- Ensures high-quality listings on the public map

Sprint 4: Enhanced Approach: These use cases further the use cases that were created for our MVP from Sprint 3. They include more personalized features, as well as additional security.

Use Case 1: Email verification unlocking personalized experience

- A new user, Alex, downloads the app hoping to keep track of food trucks near campus.
- They create an account with their email. Before accessing core app features such as saving favorites or enabling per-truck notifications, Alex must verify their email. They open their inbox, click the verification link, and are automatically re-directed into the app.
- Now, Alex can complete their profile, optionally adding their phone number so that they can receive notifications when their favorite trucks go live.
- When exploring the map, Alex sees a cleaner, more personalized interface: only relevant reporting fields show up for regular users.
- They favorite a ramen truck. Days later they receive a notification that this truck is active nearby, go to the map, and immediately find it thanks to crowd-based confirmations and UI cues about busyness.

Key features used:

- Email verification as a security gate
- Personalized profile (favorites, phone number)
- Per-truck push notifications
- Reduced reporting clutter for users
- Improved trust via verified sightings

Value demonstrated:

- Increased security
- Reduced user friction long-term
- Improved retention through personalization

Use Case 2: Vendor profile streamlines reporting and enhances discoverability

- A local vendor, Mei, owns a poke-bowl truck and wants to experiment with new lunch spots.
- She signs in through her verified vendor account and is prompted to complete her profile: truck name, cuisine type, menu, and prices. This information is saved permanently so she does not need to re-enter it every day.
- When Mei parks at her new location, she opens the app. Unlike regular users, the vendor reporting page is shorter and more efficient: she only needs to confirm location.
- The truck instantly appears on the map with a vendor-verified star. Users can tap the pin to see menu items and pricing, helping them decide whether to visit.
- Because this workflow is smooth, Mei reports consistently and gathers crowdsourced confirmations, increasing credibility.

Key features used:

- Vendor profile page stores reusable info
- Vendor-specific reporting flow (location only)
- Verified-vendor star on pins
- A/B tested pin designs

Value demonstrated:

- Eliminates repetitive data entry burden
- Improves accuracy of truck information
- Strengthens vendor–customer trust
- Encourages more consistent vendor reporting

Use Case 3: Crowdsourced validation, pin fading, and community-driven accuracy

- A college student, Priya, sees a new truck shown on the map that she hasn't heard of before. Its pin is lightly colored, indicating low confirmation. She hikes over and finds that it is indeed present; she confirms the sighting and leaves a rating.
- Over the next few hours, two more users also confirm the truck, pushing it over the required threshold of three confirmations, turning the pin more saturated.
- Later that day, after the truck leaves, fewer confirmations are reported. After several hours of inactivity, the pin gradually fades and eventually disappears from the map.
- This feedback loop helps keep the map fresh and accurate without central moderation.
- The evolving pin colors, vendor star, and rating make it easier for users to judge whether a truck is reliable and worth a trip.

Key features used:

- Confirmation threshold (3) for trustworthy sightings
- Pin visualization tied to reliability & crowd level

- Pin fade-out over time to remove stale sightings
- Reporting page slightly different for users vs vendors
- Optional rating stars

Value demonstrated:
- Maintains high-trust, self-correcting community data
- Reduces noise from unverified or outdated reports
- Encourages participation in confirmation ecosystem

Updated Selected Approach: These use cases further the use cases that were created for our Selected Approach and aims to be more tailored to our app.

Use Case 1: Sarah a young product manager with a 45-minute lunch break trying to find a nearby open food truck in time.
- She opens the app and logins
- She open up the map page where she can see verified green markers for nearby food trucks
- She presses on a marker and gets “Taco Mex, Mexican, Light Crowd and 10 mins away”
- Upon arrival, she confirms the sighting, contributing to the community verification system
- She gets and eats her tacos, and makes it back to work on time

Use Case 2: Jason a graduate student exploring the local food truck scene
- He opens the app and browses the map showing all the verified and active food trucks
- He filters by cuisine type to find new Vietnamese truck
- After pressing on the marker finds that it is only 15 mins away and has a light crowd level
- He reports the sighting and orders a sandwich
- He really like the Banh Mi Sandwich so he saves the food truck to his favorites list

Use Case 3: David a loyal customer of "Smokehouse BBQ" truck
- He opens the app and checks his "Favorites" tab
- He sees that his favorite BBQ truck is active with a verified status and 15+ confirmations today but in new location
- He sees a special "BOGO brisket" deal noted on the vendors profile
- He then navigates using the app to an unfamiliar part of town where the truck is testing new location
- He then confirms the trucks location and buys a BOGO brisket

Selected Approach: These use cases further the use cases that were created in Sprint 1’s Approach 1 and Approach 3.

Use Case 1: A young worker gets off work early and wants to try something new for dinner.

- They open the app and instantly see a map of nearby food trucks currently operating.
- They filter by "within 15 minutes walking distance."
- They find a taco truck with great reviews, see photos of the food on its profile, and navigate to the location.
- Upon arrival, they confirm the truck's presence in the app, which updates the status for other users.

Use Case 2: A new dessert truck wants to build visibility at a weekend festival.

- The vendor sets up their profile with photos of their menu and pricing
- They share their live location when they open for business.
- Nearby customers gets notified and confirms the truck's presence upon arrival
- After closing, they set it to "closed," so customers stop seeing them on the map and no longer get notified.

Use Case 3: A regular wants to know if their favorite BBQ truck is nearby for lunch.

- They open the app and go to their saved "Favorites" list.
- They see the vendor's posted schedule, but also notice that the truck is "live now" in a slightly different spot than usual but is a little far away.
- But they see that the truck is doing a limited BOGO deal and has been visited and validated by customers, so they decided to go

Approach 1:

Use Case 1: Existing User, local office worker who can't find their usual favorite food truck

- The user logs into the app with their existing account and uses the history of "recent searches" to find the listing for the food truck.
- The user can see an updated pin for the food truck in a nearby location as reported by other users not too far from them. The app displays how long it will take to reach the food truck by foot.
- The user follows the route to reach the food truck, and finds the food truck in the new location. They start to stand in line to grab a bite.
- The user also decides to confirm the sighting and rack up some points on their account.

Use Case 2: Local Food Truck Owner tired of having to constantly update social media and different apps on the location

- The owner creates a new account on the application with basic information about their food truck (similar to street food finder).

- They can optionally list their regular location(s) during business hours on the application and go live.
- After this, they don't need to take any action: as users encounter the food truck, they will report/confirm the location on the application. Now when the food truck owner decides to try out a new location or move to a new location midday, they don't need to take any action in the app.

Use Case 3: New user who is also new to the area: college student who's on the go and wants a quick bite from a food truck

- Upon clicking into the application, the first thing the user sees is the map with pins for different food trucks around their location.
- Due to time rush, the student selects a location filter to find food trucks that are within a five minute walk from them to see which food trucks are close by.
- The student sees a food truck rated at 4 stars that's 2 minutes away from them. They confirm their selection and begin to follow the route the app has calculated.
- The student reaches the food truck and successfully obtains their quick bite.
- Happy with the experience, the user creates an account on the application with basic information (email, username, location permissions, preferences), confirms the sighting and logs points.

Approach 2:

Use Case 1: Food truck owner preparing for the lunch rush.

- The vendor parks at their designated lunch spot downtown.
- They open the vendor tool on their phone and input the truck's location (typing location or GPS autofill).
- The vendor presses "Post Update."
- The app simultaneously pushes the update to social media platforms, like Instagram and Twitter, that are associated with the vendor's account. The message is preformatted including the truck's name, location, and serving hours at this location.
- Customers following the truck on social media instantly see the post and begin arriving for lunch.

Use Case 2: Midday Change of Location

- The vendor wants to move from their current spot due to time limit restrictions at this location. They relocate to a new area near a college campus.
- On arrival, the vendor reopens the tool, selects "Update Location," and confirms the new GPS-tagged address.
- The app automatically generates a new social media post for all associated accounts with the truck's name, new location, and serving hours at this new location.
- Followers see the update in real time, and students nearby stop by for dinner.

Use Case 3: Food truck owner finishing service after a busy day.

- The vendor realizes that they are sold out and have to close earlier than they posted.
- They open the vendor tool and select a “status update” option to update to “sold out - closing early”. This generates a social media post accordingly.
- Followers see this update in real time, preventing wasted trips from customers who might otherwise show up after closing.
- The vendor avoids disappointing late-arriving customers while maintaining transparency.

Approach 3:

Use Case 1: A college student wants to eat out with friends and is tired of the same places but isn’t sure of where to go.

- The student opens the app to the main map page.
- They can then see any food trucks currently operating in the area
- When they press on the food truck, they can see information about it
- After choosing which food truck they want to go to, the truck's current coordinates are given and the user can use that to map to the location using another navigation app like google maps or apple maps.

Use Case 2: A regular wants to know where is favorite food truck is, to take his friend

- The regular opens the app and goes to his saved list
- He then searches up his favorite vendor
- After clicking on the vendor, a profile with the vendors schedule is opened up saying that it is currently not operating
- Alternatively if the vendor is currently operating then the location on the map is given after pressing on them

Use Case 3: A new vendor wants a way to gain more customers and downloads the app

- The vendor hears about this app that will give his location to nearby customers and decides to download it
- After downloading, the app asks him to make a profile where he can optionally upload photos, give a description of his business and background, and set up a schedule of regular operation hours
- Once he is done, he decides to update his status to currently operating
- Nearby customers see his location and heads over

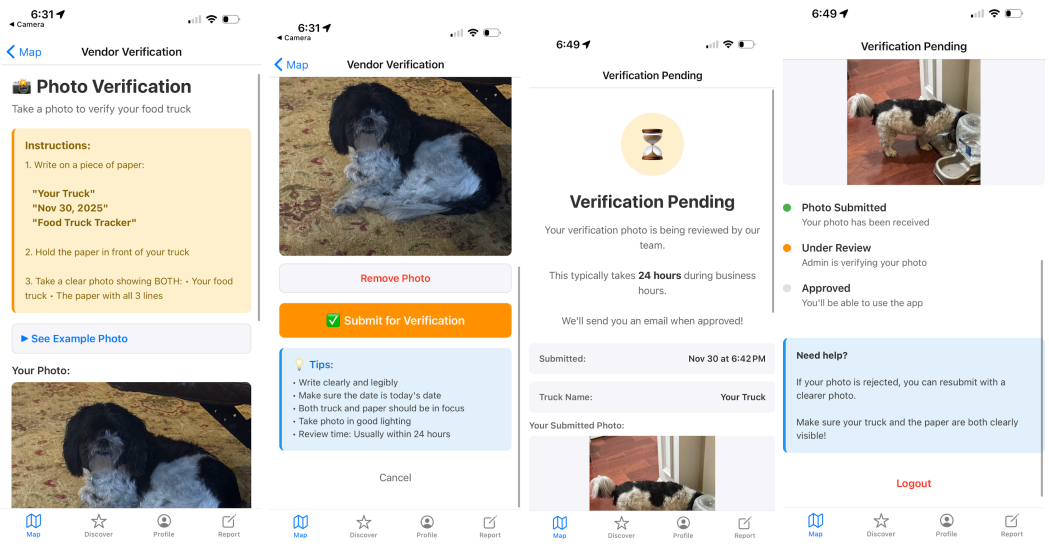
Learning Prototype

Sprint 5 Learning Prototype:

In Sprint 5, based on peer feedback and our own concerns with overfitting the application to users, we used this final sprint to focus on testing the application with vendors to help us round out our MVP before the semester ended. In our interviews and testing rounds, one thing we continuously came back to was trust and accuracy: if users didn't trust the food truck locations posted on the application (i.e., they felt a food truck posting was fake), it wouldn't matter if the data was updated in real time. We already had several accuracy and trust-focused features developed from the user perspective, so from the vendor perspective, vendor verification seemed like a natural next step. However, we were unsure what form this process should take so that it didn't negatively impact vendors' desire to use the application. In developing this learning prototype, then, we were essentially looking to get more clarity on how to introduce a process to make sure that data supplied from the vendor perspective is trustworthy without unwittingly pushing vendors away.

As such, we tested two different versions of a photo-based verification feature with two different groups of vendors. Based on TA and peer feedback, rather than adding a bulk load of features, we had both vendors go through the same process, with one key difference: testers in group A could not access any parts of the application (besides the map) unless they went through the verification process, while those in group B did not face this restriction.

Verification Screens (Common across both groups):

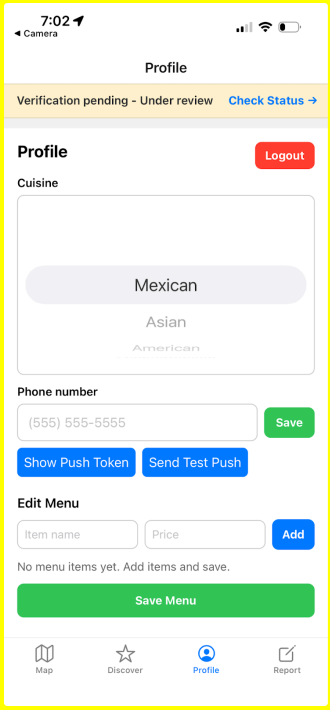


These screens from left to right show how the verification process works. Essentially, to be verified, a vendor must upload a picture of both their truck and a paper in front of it that states the name of their truck, the current date, and “Food Truck Tracker”. They can either select this from their gallery or take the photo in the app itself (a sample photo of a dog was taken for demonstration’s purpose only of how an uploaded photo would look in the app). Upon submission, vendors will be redirected to a page that displays their verification status so they can keep track of where they are in the verification process. In the meantime, administrators will review the submission and approve the vendor's uploaded picture in Firebase itself to denote that the vendor has been verified.

Version A:

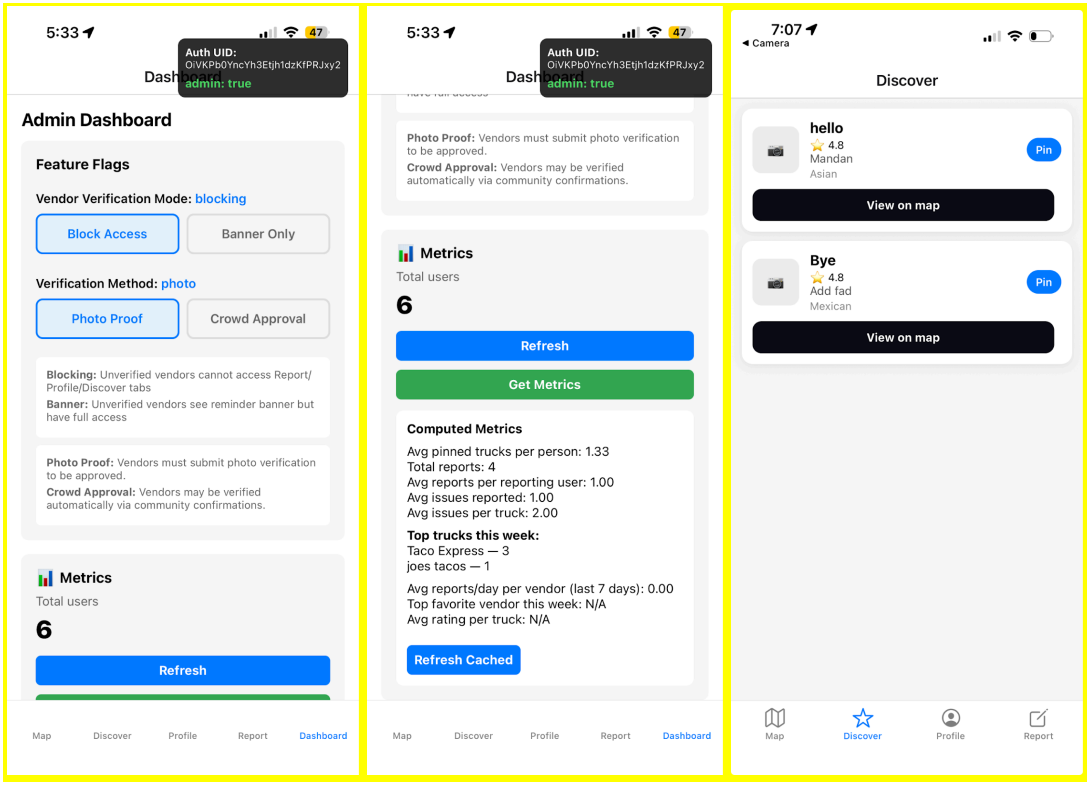
No changes were made to the current UI. Once verification is complete and approved from the admin side, vendors can use the app as normal.

Version B:



Instead of blocking vendors from using the app if they’re not verified, Version B of the application simply displays a banner at the top for vendors. If the vendor has not submitted any photo for verification the banner instead reminds vendors to go through the verification process with the message “Get verified for more credibility with potential customers!”. Upon being successfully verified by the admin, the banner is no longer displayed.

Additional Features: Dashboard & Discover Page



The first two screenshots show the dashboard developed for admin use only that not only displays key user metrics (see the second screenshot) but also allows admin to control which version of the vendor verification feature is currently active during testing. Note that the first two screens in particular were not tested themselves but instead used to assist in testing and to fulfill the second platform requirement.

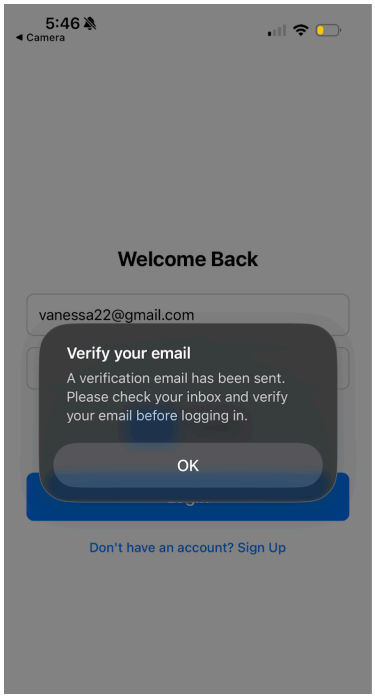
The third screenshot is a screenshot of a discover page that features and promotes select trucks. As our app viability partially relies on advertising, the other key question we wanted to answer before closing off the course was how much perceived value vendors would see in in-app advertising and would this impact their willingness to adopt the application. Vendors were shown this screen and asked questions regarding their willingness to pay for in-app promotions (as a note: no promotion request or advertising payment services themselves were implemented). All questions used in and results collected from testing are linked and discussed in the Learning Prototype Results section.

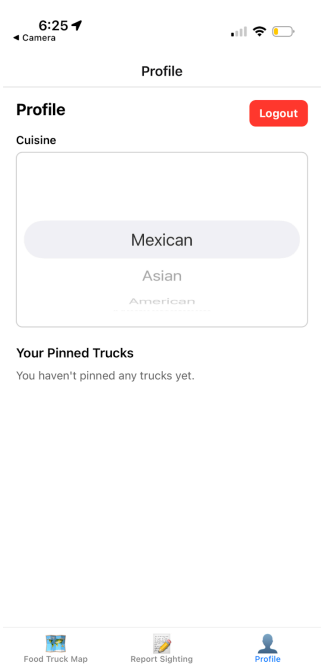
Sprint 4 Learning Prototype:

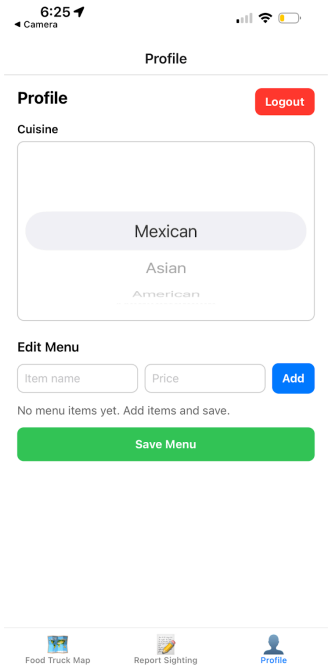
In sprint 4, we began working on enhancements for the application beyond the main functionality so that our different user types (users and vendors) would each have a tailored experience. Based on the feedback we received in class discussions, we also had new questions and went into developing this prototype with the goal of testing what threshold of confirmations would increase users trust in our application, what information about food trucks users would find most useful to encourage information with this sort of application, and whether or not a minimalistic presentation of that information would encourage or discourage engagement.

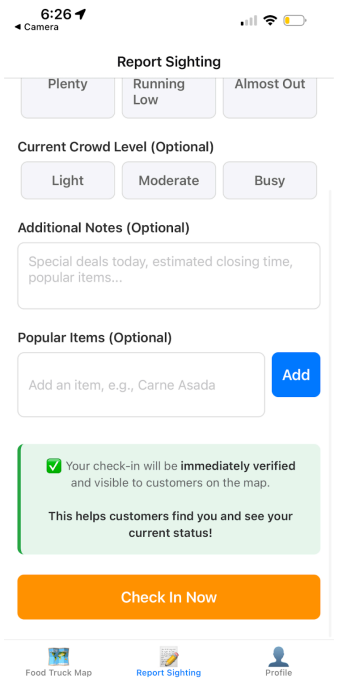
As such, we tested two different versions of our planned pin enhancement with two different groups of users. Although all other features added were common between testing, the modal displayed when clicking on a food truck was different for these two groups.

Added Features Common to Both Versions:





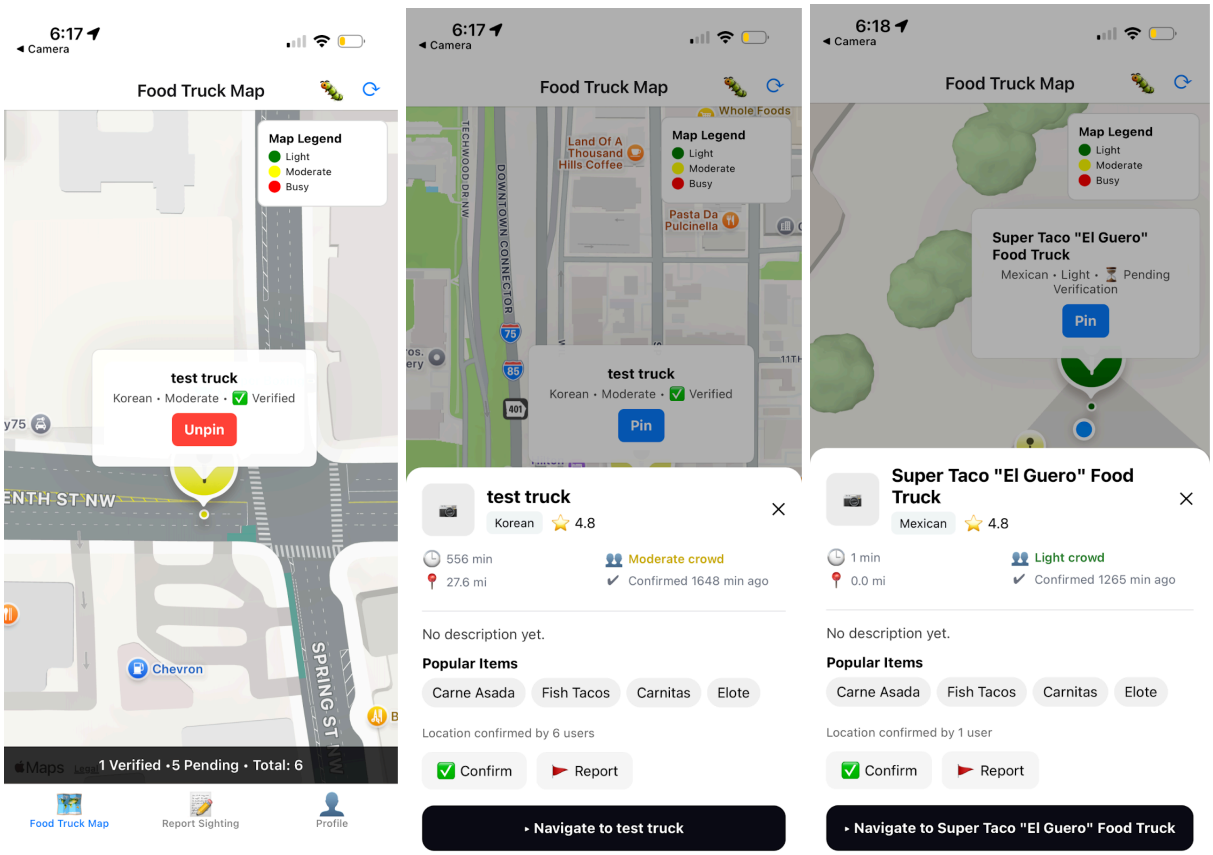




These screenshots show the features that we developed for both versions of the application, regardless of testing in order to address feedback and suggestions given in class discussions and to increase general trust of the application and provide a more tailored experience. First, we implemented email verification so that fake emails could not be used to create an account on our platform with the goal of increasing the accuracy and authenticity of food truck sightings. We

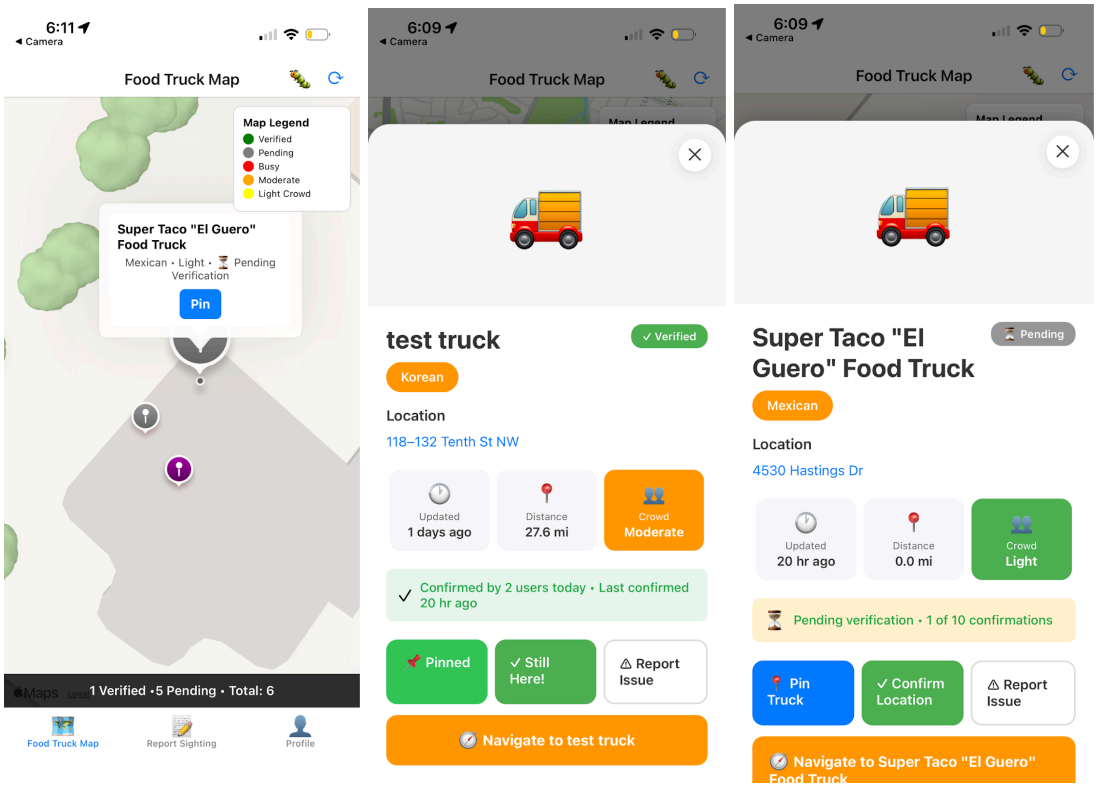
also developed a tailored profile page for each user type (the left for users and the right for vendors). Users could pin trucks they wanted to keep track of and vendors could save information about their food truck in the application that they wanted to display. We previously had a general “Report Sighting Page” for users only (refer to Sprint 3 Learning Prototype). In this sprint, we also added a version of the page visible to vendors only where they could “check-in” and update their location as well as update food inventory and any special offerings.

Version A:



Version A features a modal (in the middle and right images) with minimal colors and components but features extra information that Version B does not include. Namely, popular items and a displayed rating were also included. Version A’s modal also does not offer users the ability to pin the food truck. That option was instead only displayed for users before clicking into the modal. The threshold required for a food truck sighting to become verified in Version A was 3 (same as last time).

Version B:



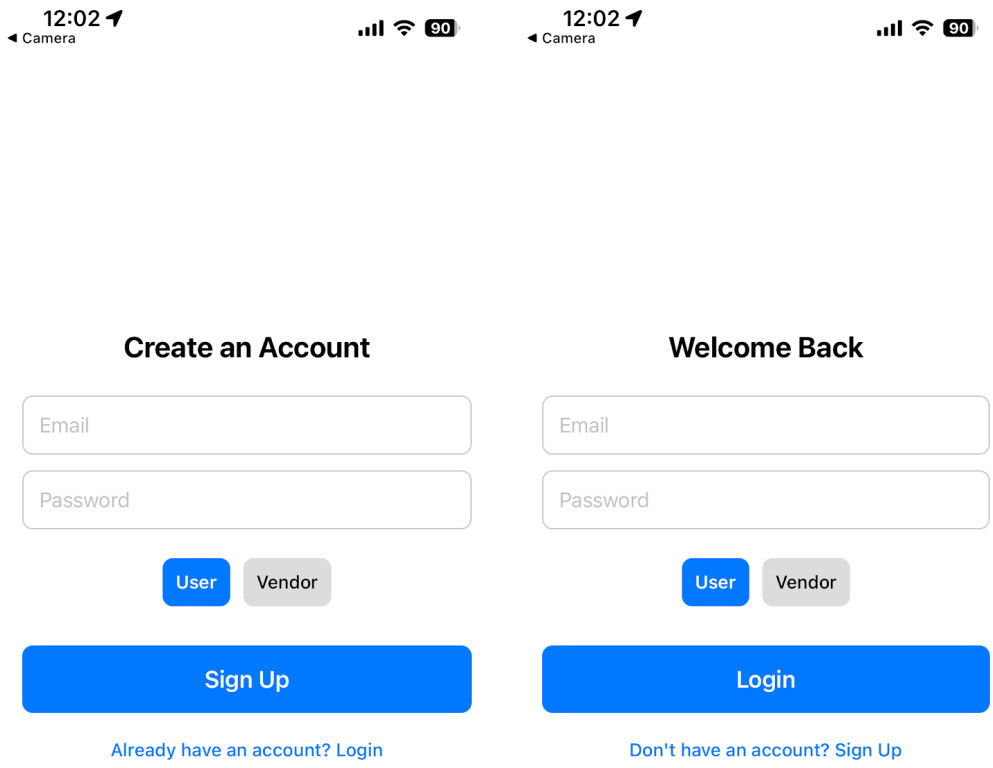
Version B of the application features a more colorful modal which offers the same information as Version A. The main differences here are that Version B's modal omits the details on ratings and food items, includes the option to pin the truck, and also displays the number of confirmations still needed for the truck to become verified. For Version B, that threshold was increased to 10. Essentially, while Version B was less minimalistic and more colorful, it displayed less information to the user.

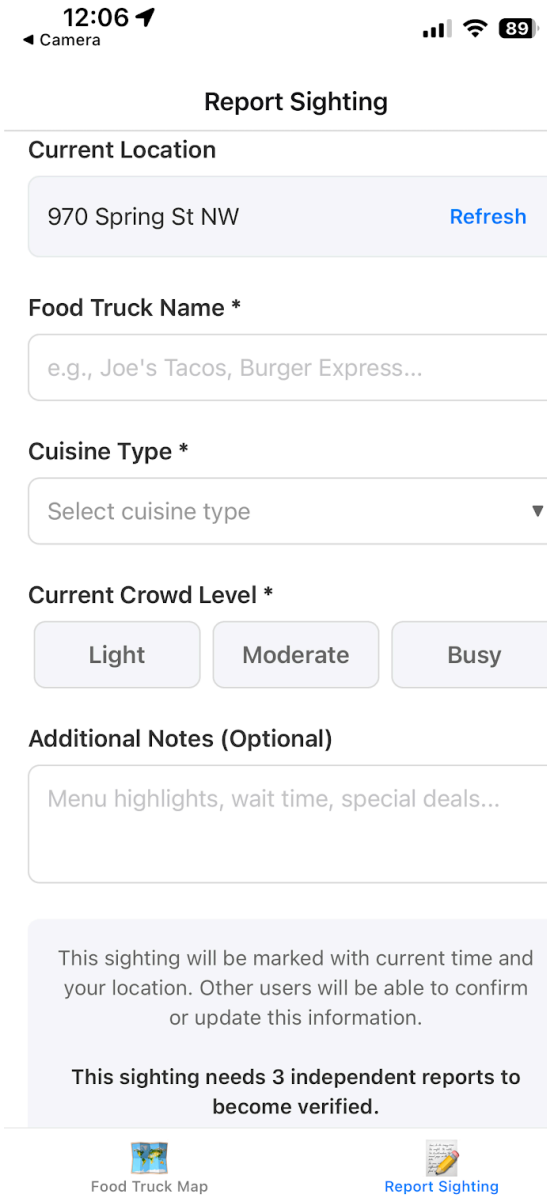
During each of our tests with different user groups, while some questions were specific to the version displayed we mainly asked the same questions to ensure consistency in testing and so that we could effectively analyze any differences in the answers to the questions and discuss implications. These common questions are linked below.

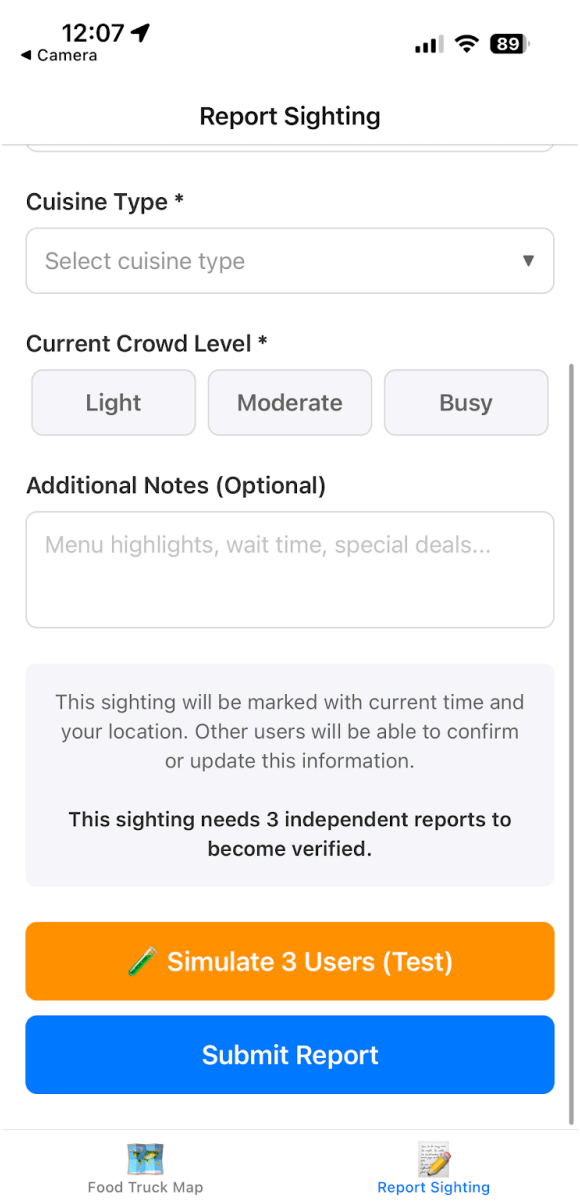
Sprint 4 User Testing Questions

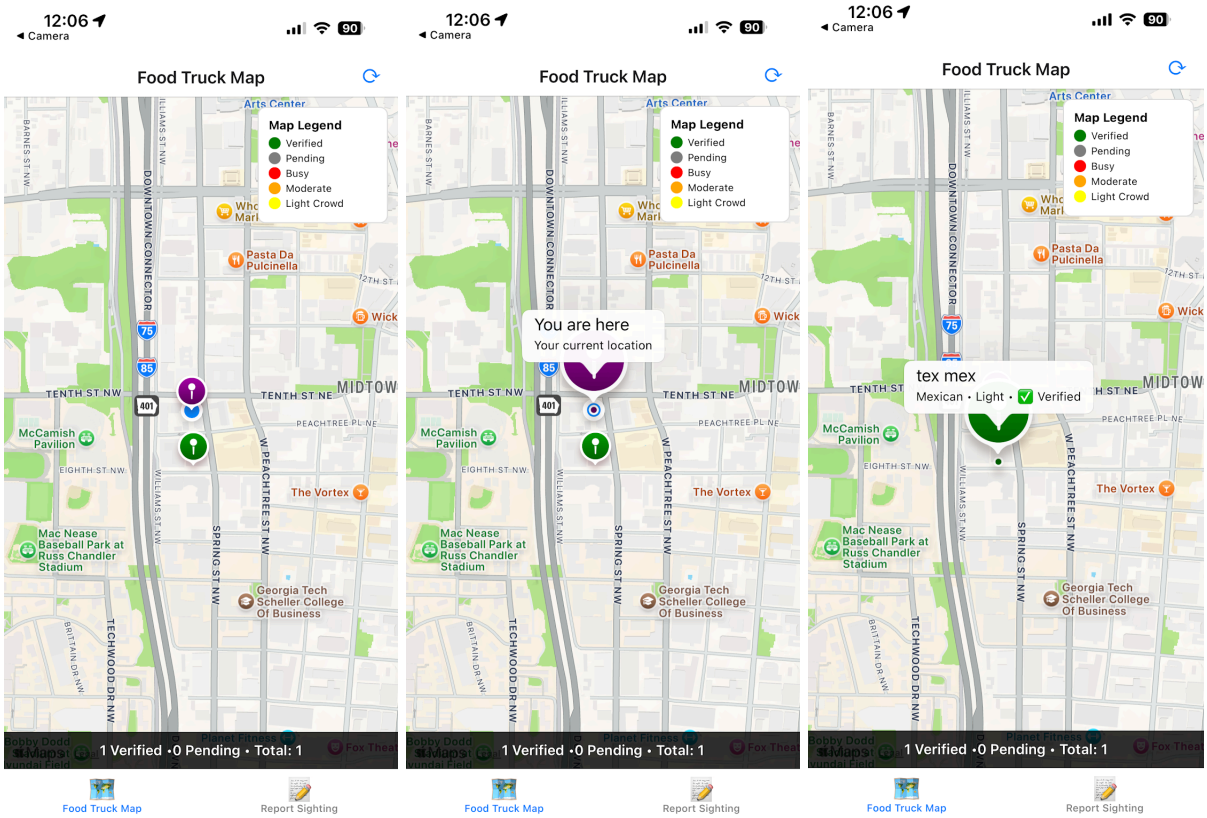
Sprint 3 Learning Prototype:
In Sprint 3, we began developing the app. We created a sign-up/log-in page where app users can input their email, password, and select whether they are a user or vendor. We also created a map

page and a reporting page. The map page has a legend to show what types of pins are reported (vendor reported vs. user reports with crowd level). It also shows your current location. The reporting page allows users and vendors to input the name of the truck, type of food, crowd level, and additional comments. For user reporting, we added a simulation so that one user could report three times in order to have the pin appear on the map. This was done solely for testing purposes. Vendor reporting still uses the submit button.



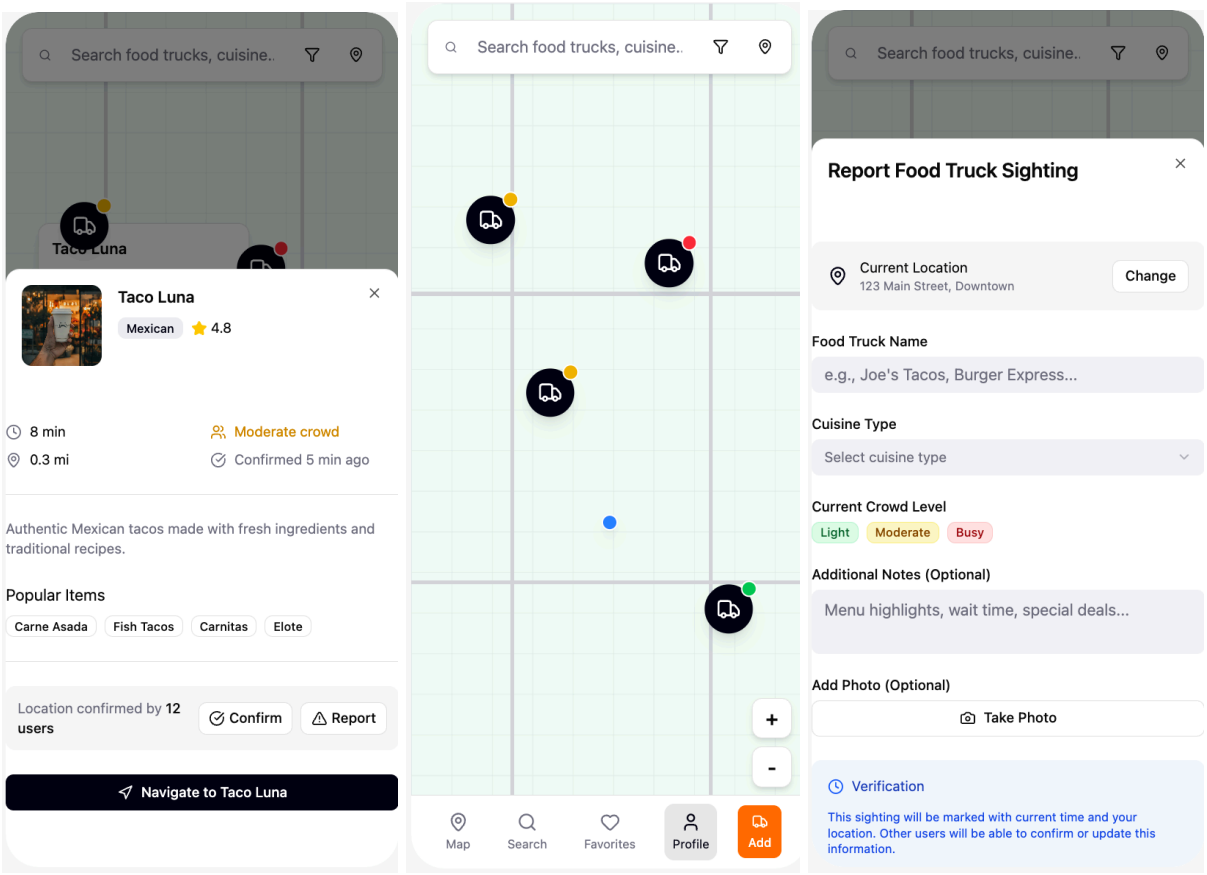

12:06
Camera
89
Report Sighting
Current Location
970 Spring St NW
Refresh
Food Truck Name *
e.g., Joe's Tacos, Burger Express...
Cuisine Type *
Select cuisine type
Current Crowd Level *
Light
Moderate
Busy
Additional Notes (Optional)
Menu highlights, wait time, special deals...
This sighting will be marked with current time and your location. Other users will be able to confirm or update this information.
This sighting needs 3 independent reports to become verified.
Food Truck Map
Report Sighting


12:07
Camera
89
Report Sighting
Cuisine Type *
Select cuisine type
Current Crowd Level *
Light
Moderate
Busy
Additional Notes (Optional)
Menu highlights, wait time, special deals...
This sighting will be marked with current time and your location. Other users will be able to confirm or update this information.
This sighting needs 3 independent reports to become verified.
Simulate 3 Users (Test)
Submit Report
Food Truck Map
Report Sighting


12:06
Camera
Food Truck Map
Map Legend
Verified
Pending
Busy
Moderate
Light Crowd
1 Verified • 0 Pending • Total: 1
Food Truck Map
Report Sighting
12:06
Camera
Food Truck Map
Map Legend
Verified
Pending
Busy
Moderate
Light Crowd
You are here
Your current location
1 Verified • 0 Pending • Total: 1
Food Truck Map
Report Sighting
12:06
Camera
Food Truck Map
Map Legend
Verified
Pending
Busy
Moderate
Light Crowd
tex mex
Mexican • Light • Verified
1 Verified • 0 Pending • Total: 1
Food Truck Map
Report Sighting

Sprint 2 Learning Prototypes:
In Sprint 2, we developed learning prototypes for each of the three solution approaches to visualize each of their key features from their primary user perspectives and make a better determination on what our chosen approach should be.

Approach 1:
Figma mockup for crowdsourced map of food trucks, page to report a food truck and food truck listing information. This is *not* clickable, instead it just captures isolated screens to capture the key features of the approach.



A customer using the application can look up a food truck they are interested in and see relevant information including wait times and proximity as well as popular items. They see a map that lists food trucks close by with the colored circle indicating the current crowd level. The user can also optionally contribute to the app's data by confirming the location of a food truck, reporting inaccuracy or by even reporting a new food truck sighting themselves.

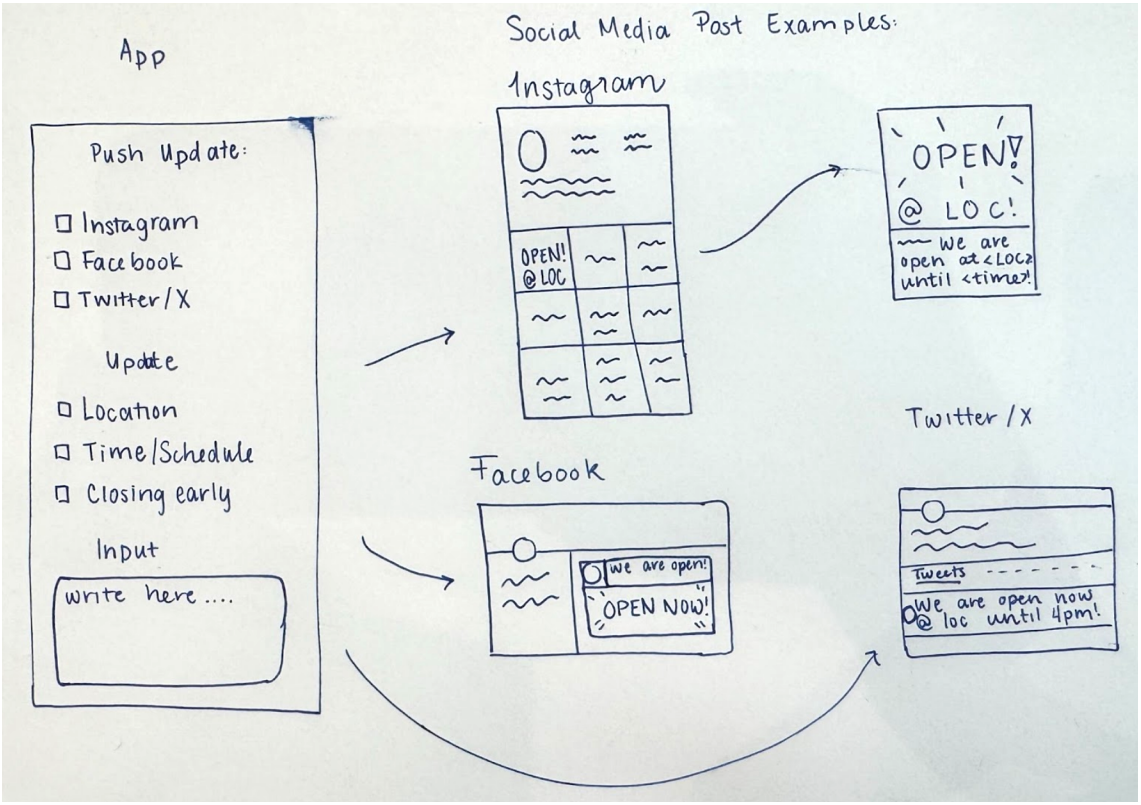
A small online survey targeted towards food truck customers and vendors intended on gauging interest and need for the application as well as likelihood of regular usage was also released both based on our learning prototype plans and based on classmate recommendations to gather more concrete data on the viability of the approach and consider potential features. The link to the

survey questions is included below. The link to the results of this survey and its implications will be discussed as a whole with the results from the other learning prototypes in the "Learning Prototype Results" section later in this notebook.

[Survey: Food Pop-up & Food Truck App Survey - The Rolling Crew]

Approach 2:
To test approach 2, two ideas were created. The first was to create a mock demo of the app, from inputting a location to generating a post across social media. The second testing idea was to create a prototype and partner with a vendor to test out the app. Based on their experience, we would refine the idea. However, this is a step that should be done later, after we decide what approach to continue forward with. This is something to keep in mind as we build our app and continue to refine it through testing with different user groups.

Below is the low-level mock-up of the app. This was done with a paper design, as recommended in lecture, to give people a basic idea of the functionality of the app, rather than focusing on design and UI. The mock-up gives vendors the option to select which social media platforms to push an update to, as well as what the update is. It also lets them upload extra information for the update. The right side shows how messages or updates will show up directly on their social media feed in their account as the most recent post.

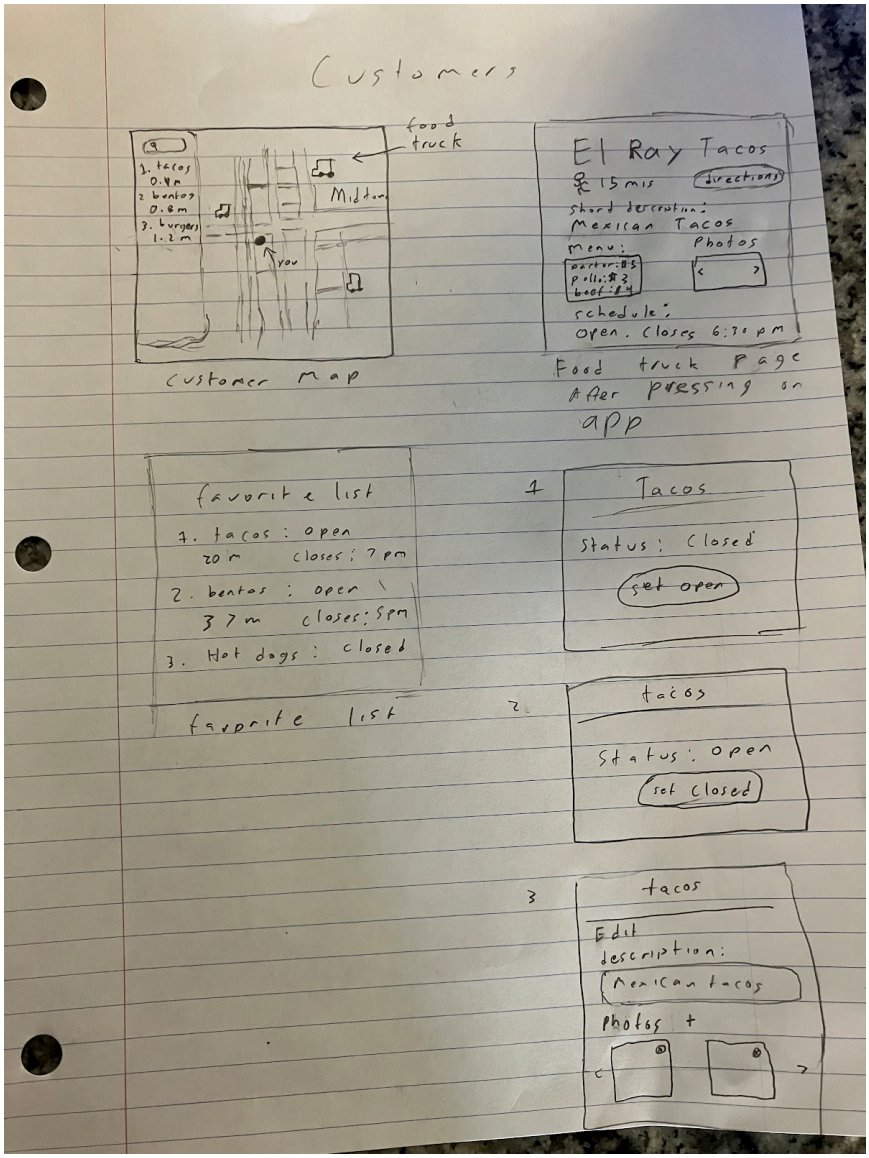


Approach 3:

For approach 3, we decided to test our concept by creating a mock up of the customer facing side and a story board of the vendor facing side. Both the mock up and story board are meant to be simple and showcase the functionality of the approach, so the UI hasn't been refined and is intentionally kept simple.

For the customer mockup we drew out the map showing nearby food trucks, shown with the truck icon, that they could click on to take them to the vendors/trucks profile. On the left of the map they have a favorites list that shows them which trucks are active, how far they are and when they will close. For the vendor/trucks profile page that users can see, it shows the walking time and distance, a short description of the food truck, a picture of their menu and items, if they are open and when they close. On the page customers can click on the directions to navigate to the current vendor's location.

For the vendors, I drew out a storyboard showing how they would set up their profiles and what information they had to fill out. I also showed how they would update their status from active to not active and vice versa.



Sprint 1:

Approach 1:

1. Would vendors see enough convenience with listing themselves on an application where they don't need to manually update their location? Would they rather have more control and privacy?
2. Given that many consumers who eat at food trucks do so on spur-of-the-moment, how many regular consumers vs. foodies are likely to use the app *regularly* and update the app as it requires?
3. Is there any way this sort of application could be extensible to smaller cities? For events, festivals etc.?

4. What types of information (wait times, ratings etc.) are most valuable for users to track on this sort of app besides locations and routes?
5. How would we prevent the problem of having the map be empty early on when the user base is small? How would we set up early user adoption?

Objective: Determine whether or not an application like this could be regularly used by a variety of users in different locations and what are features of importance for all user types to encourage regular use of the application.

Hypothesis: Food truck owners in bigger cities will see more value in not having to engage regularly with this app, while actual customers are more likely to see value if they engage with food trucks regularly and can receive a discount or incentive to report food truck locations.

Testing:

- To determine whether this solution is viable, we can conduct a survey across Georgia Tech or Atlanta, showcasing mockups (this would be built both for the survey but also to get us thinking about feasibility) for what the potential application could look like at a high level and ask questions related to how likely users are to engage with it and how often.
- We can also ask users to rank what sorts of features would be most important to them in order to use this sort of application. This way we can collect large scale quantitative data that gives the information needed to confirm or deny the hypothesis.

Approach 2:

1. Will vendors actually use a separate tool to post updates if it only benefits them indirectly through customer convenience?
2. How often would vendors update their social media during busy hours?
3. What kinds of location/status updates are most important for vendors to share with customers?
4. Do vendors see value in auto-posting features, or would they worry about losing control over the wording or branding of their social media posts?
5. Which social media platforms are most important for vendors to integrate with?

Objective: Validate whether vendors see enough value in a single-post-to-multiple-platforms tool to adopt it in their daily workflow.

Hypothesis: Vendors will adopt an automated social media updating tool if it saves them time, allows simple customization, and keeps their updates consistent across platforms.

Testing:

- We can create a mock demo showing how food truck vendors would input a location. It would show what would be generated and posted across their platforms. This would help gather feedback on simpler and most wanted features.
- Eventually, once we have a prototype, we can partner with a vendor for a week and test out the app. We can track whether they post more consistently and the impact customers have, such as bigger crowds and more platform engagement.

Approach 3:

1. Will vendors be willing to download and input their information, and updating status, i.e. will they find it inconvenient?
2. How close do vendors have to be in order for customers to be willing to go?
3. Will vendors forget to update their status causing customers to go to them when they are already closing.
4. How far away should a vendor be to notify customers, because if there are too many vendors, then customers can feel overwhelmed by notifications?
5. What information will customers want to know about vendors before deciding to go to the truck?

Objective: Find out which features are high value (will make vendors and customers want to adopt) vs which features will deter vendors and customers from using the solution/app. What features are of higher priority.

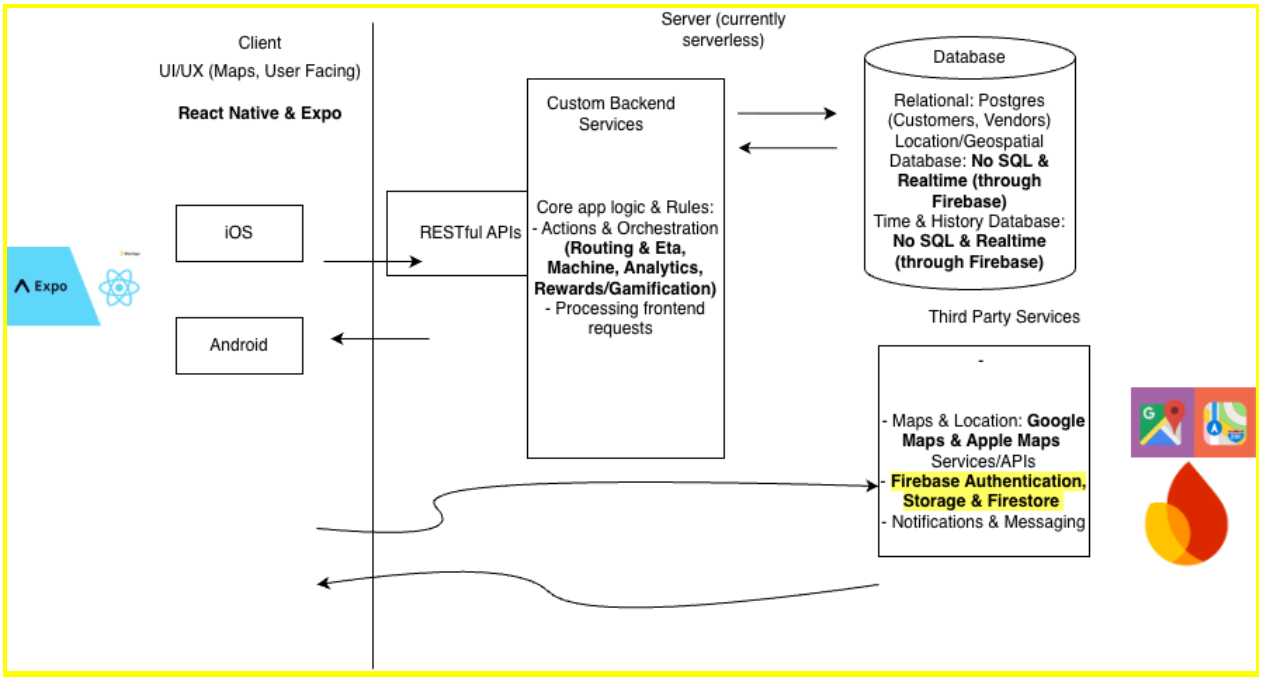
Hypothesis: Food truck vendors will use the app to promote business to customers so long as the app isn't too inconvenient to use, while customers will use the app if there are a lot of vendors and if they aren't spammed with too many notifications.

Testing:

- On the customer side we can create a mock up of different features such as a real time map where customers can see nearby vendors or a saved list of vendors that will notify users when operating, to see which features have the most appeal.
- On the vendor side, we can create a mock where they create their personalized profiles, update their status, etc; in order to see which features act as the biggest barrier of entry or of continued use.
- We could also send out a survey to get user opinions. This would give us an idea of what features users would want and which features they care less about.

Technical Discussion

Sprint 5:



As of Sprint 5, our final MVP for the course still follows a serverless architecture. However, since the last sprint, we've expanded this architecture to include vendor verification and admin dashboard infrastructure while maintaining the core functions of the system.

Frontend Platform and Tools

Our learning prototype remains serverless and uses React Native with Expo for cross-platform mobile development (iOS & Android). We built our frontend using:

- React Native and Expo for rapid prototyping, live reload, and native sensor access (GPS, camera).
- react-native-maps for the live map interface, rendering both Apple Maps (iOS) and Google Maps (Android).
- Expo Location API to access the user's live GPS coordinates.
- AsyncStorage for lightweight local caching of user data (e.g., email, userType).
- Firebase SDK (modular v9) for real-time database interaction and authentication.

The client either directly calls native APIs (Location, Camera) or interacts with SDKs (Firebase, Google Maps), which handle all REST API calls internally to Firestore.

Backend Platform (Firebase)

Since the beginning of development, our team has utilized Firebase Firestore as the database and backend service for our application. This allows our client application to directly read and write to JSON documents without the need for maintaining a custom backend (no custom REST API or server). Firebase Authentication is used for managing user login sessions (user vs vendor). Firestore's real-time listeners (onSnapshot) automatically push updates to all clients, so users can immediately see newly reported or verified trucks without reloading. In this last sprint, the key change (highlighted in the diagram) is that we additionally enabled Firebase Storage to store photos uploaded by vendors for vendor verification purposes, with security rules enforcing authenticated uploads.

Each food truck sighting is stored as a Firestore document:

json

```
{
  "foodTruckName": "Yumbii",
  "cuisineType": "Korean-Mexican",
  "crowdLevel": "Moderate",
  "status": "verified",
  "timestamp": "2025-11-09T03:00:00Z",
  "location": { "latitude": 33.776, "longitude": -84.389 },
  "favoriteItems": ["Bulgogi Tacos", "Kimchi Fries"],
  "additionalNotes": "Behind Tech Green today",
  "reporterId": "user@example.com"
}
```

Key Implemented Logic

This logic, we've slowly built up through the entirety of development, and it stands from the last sprint:

1. Verified Pin Filtering: Same as before, the map displays only trucks marked as verified. This allows us to ensure data reliability and accuracy for users.
2. Color-Coded Pins: The pins displayed on the map for a food truck are colored differently based on crowd level: green (light), yellow (moderate), red (busy)
3. Automatic Cleanup: A background routine (clearOldSightings()) deletes non-verified reports older than 24 hours to prevent stale data.
4. Food Truck Detail Popup: When a marker is tapped, the app shows a food truck detail card as a bottom sheet with key information such as distance and ETA, as well as the user-reported popular items, crowd level, last confirmed time, average rating, and buttons for the user to confirm or report issues with the sighting.

5. Favorites / Pinned Trucks: Users can tap “Pin” to store favorite trucks under the favorites collection in Firestore, linked by user email.

Since the last sprint, we’ve also made key additions to our application logic.

6. Vendor Verification: In this last sprint, we added a photo-based system that requires truck owners to submit images for admin verification to help prevent an inauthentic food truck from posing as authentic on the application. Admins review in Firebase Console and approve or reject submissions.
7. Issue Reporting: We have now implemented our issue reporting feature to help us ensure the reliability of the information on our application. Users report incorrect locations or closed trucks to the reportedIssues collection for admin review
8. Push Notifications: Expo Push Notifications alert subscribed users when favorited trucks check in nearby

Data Flow

1. User -> Firestore:
   When users report or confirm a food truck, the app writes data directly to the sightings collection via Firebase SDK.
2. Firestore -> All Clients:
   The real-time listener in MapScreen immediately updates all connected devices with new or updated truck data.
3. Device Sensors -> UI:
   The app uses Expo’s Location API to dynamically center the map around the user and compute distances to trucks.

No external web APIs are called directly: both maps and location APIs are managed through native SDK wrappers.

Implemented REST-Like Data Operations

Although Firebase abstracts away explicit REST endpoints, the equivalent logical operations have been included below. For conciseness, we’ve omitted the previous REST-like data operations that were mentioned in the last sprint. They are still utilized in the application and can be referenced in the previous section for Sprint 4. The new additions are as follows:

Operation	URI/Collection	HTTP Verb	Description

/vendors/:email	Firestore	GET	Fetch vendor profile & verification status
/verification_photos/*	Storage	POST	Upload vendor verification photo (blob)
/reportedIssues	Firestore	POST	Submit an issue report if any incorrect truck data is being displayed on the map
/analytics	Firestore	POST	Log user events for analysis
/userSubscriptions/:userId	Firestore	PATCH	Manage push notification subscriptions
/users/:uid	Firestore	GET	Check admin status & load user preferences

Dashboard (2nd Platform Requirement)

Although the dashboard functions are included in the same GitHub repository, we're highlighting this feature in a separate section as it satisfies the second platform requirement.

We've included a dedicated DashboardScreen that provides admin-only analytics and moderation tools. Instead of relying on paid cloud functions, our app instead uses a CI-generated metrics pipeline which essentially allows the reading of Firestore collections and computation of statistics through a Github Actions job sign-in. Results are then written to the cached document adminCache/metrics through fetchCachedMetrics(). The dashboard also allows admin to set feature flags (i.e. controlling which version of verification vendors go through for testing).

As seen in the learning prototype, our current implementation of the dashboard shows the following metrics.

- **User Metrics**: Total user count with manual refresh
- **Engagement Analytics** (computed on button press):
  - Average pinned trucks per user
  - Total reports and average reports per user
  - Average confirmations per sighting
  - Top 3 most-reported trucks (last 7 days)
  - Average reports per vendor per day

- Vendor ratings aggregation

Access Control: The users/{uid} document includes an isAdmin boolean flag checked at login, which ensures that non-admin users cannot access the Dashboard tab (conditionally rendered in the tab navigator).

For future development we've in particular left off the development of a truly custom backend (as mentioned) as well as the inclusion of rewards/gamification (see reasoning in Learning Prototype Plans for Sprint 5) and complex machine learning algorithms. This allows us to keep our learning prototype truly minimal and lightweight both for testing purposes so that we can effectively pinpoint the causes behind user behavior in the platform and so that we can avoid overwhelming users before building up a user base on our core features.

The link to our team's code repository is included here: Food Truck Tracker Github Repository. The two versions of our application are in the same branch and turned on and off through the dashboard. While the overall architecture did not change significantly, these updates capture, in greater detail, the types of operations used and the flow of data as we've added more features and goes further into detail on how the dashboard works.

Sprint 4:

Our Sprint 4 learning prototype is a fully serverless mobile application built using React Native with Expo for cross-platform support (iOS & Android). The focus for this sprint was on improving the core user flow: allowing users to report, view, and confirm food truck locations in real time, while automatically cleaning up old data and surfacing only verified trucks on the map.

Frontend Platform and Tools

We built the frontend using:

- React Native and Expo for rapid prototyping, live reload, and native sensor access (GPS, camera).
- react-native-maps for the live map interface, rendering both Apple Maps (iOS) and Google Maps (Android).
- Expo Location API to access the user's live GPS coordinates.
- AsyncStorage for lightweight local caching of user data (e.g., email, userType).
- Firebase SDK (modular v9) for real-time database interaction and authentication.

Backend Platform (Firebase)

The app uses Firebase Firestore as its database and backend service. This allows the client app to directly read and write structured JSON documents without needing a custom REST API or

server. Firebase Authentication is used for managing user login sessions (user vs vendor). Firestore's real-time listeners (onSnapshot) automatically push updates to all clients, so users can immediately see newly reported or verified trucks without reloading.

Each food truck sighting is stored as a document in the Firestore sightings collection with fields like:

```
{
  "foodTruckName": "Yumbii",
  "cuisineType": "Korean-Mexican",
  "crowdLevel": "Moderate",
  "status": "verified",
  "timestamp": "2025-11-09T03:00:00Z",
  "location": { "latitude": 33.776, "longitude": -84.389 },
  "favoriteItems": ["Bulgogi Tacos", "Kimchi Fries"],
  "additionalNotes": "Behind Tech Green today"
}
```

Key Implemented Logic

1. Verified Pin Filtering:
   The map now displays only verified food trucks (status = "verified") to ensure data reliability.
2. Color-coded Pins:
   Pins dynamically change color based on crowd level: green (light), yellow (moderate), red (busy).
3. Automatic Cleanup:
   A background routine (clearOldSightings()) deletes non-verified reports older than 24 hours to prevent stale data.
4. Manual Debug Delete (Developer Tool):
   Admins can delete individual trucks by name or all reports for debugging purposes directly from the debug panel.
5. Food Truck Detail Popup (Bottom Sheet):
   When a marker is tapped, the app shows a food truck detail card with:

- Truck name, cuisine type, and placeholder photo
- Average rating (static 4.8 for prototype)
- Crowd level, last confirmation time, and walking distance/time from user location
- User-reported popular items (aggregated from Firestore reports)
- Buttons for “Confirm,” “Report,” and “Navigate” (opens Apple/Google Maps)

6. Real-time Updates:
   The onSnapshot Firestore listener continuously synchronizes all active trucks without manual refresh.
7. Favorites / Pinned Trucks:
   Users can tap “Pin” to store favorite trucks under the favorites collection in Firestore, linked by user email.

Data Flow

1. User -> Firestore:
   When users report or confirm a food truck, the app writes data directly to the sightings collection via Firebase SDK.
2. Firestore -> All Clients:
   The real-time listener in MapScreen immediately updates all connected devices with new or updated truck data.
3. Device Sensors -> UI:
   The app uses Expo’s Location API to dynamically center the map around the user and compute distances to trucks.
4. No external web APIs are called directly: both maps and location APIs are managed through native SDK wrappers.

Implemented REST-Like Data Operations

Although Firebase abstracts away explicit REST endpoints, the equivalent logical operations are:

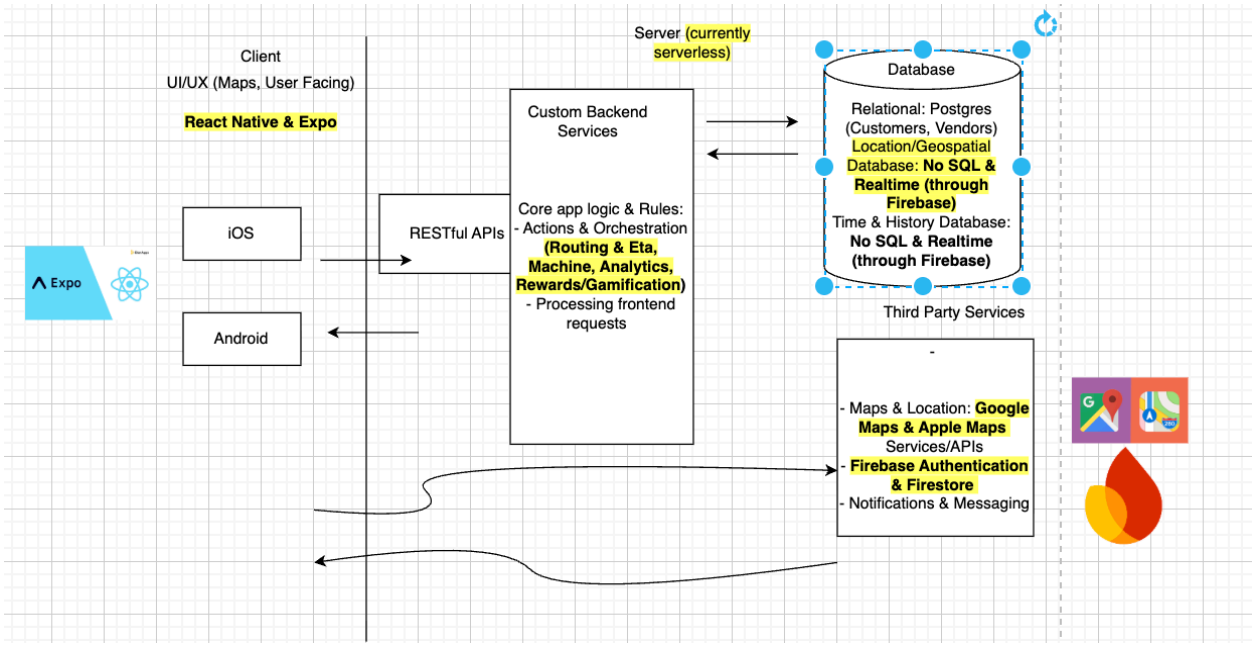
Operation	URI/Collection	HTTP Verb	Description
/sightings	Firestore	GET	Fetch all current truck sightings (real-time listener)
/sightings	Firestore	POST	Add a new truck report

/sightings/:id	Firestore	PATCH	Update truck status (e.g., verified)
/sightings/:id	Firestore	DELETE	Delete an outdated or invalid sighting
/favorites/:userEmail	Firestore	PATCH	Add/remove favorite truck for user

Our Sprint 4 app aims to be a serverless learning prototype. The technical focus was on data persistence, verification, and real-time synchronization: allowing for food trucks to appear, expire, and update automatically without manual reloads or external servers.

The link to our team's code repository is included here: [Food Truck Tracker Github Repository](). The two versions of our application are in separate branches. While the overall architecture did not significantly change (so no update to the diagram), these updates capture in greater detail, the types of operations used and the flow of data.

Sprint 3:



During Sprint 3, we revised the diagram to capture the architecture given the state of our current learning prototype. The key changes are as follows:

Our first learning prototype is **serverless**. For user reporting and authentication, we utilized an existing backend service, Firebase, which allowed us to build these features quickly without maintaining a custom backend (yet) and leverage the built-in syncing, security, and scaling functions.

For the frontend, we used React Native with expo to support cross-platform mobile development. Our current learning prototype does not support web development and is not needed to gather data from users. However, we will leave room to expand to web platforms if user interest justifies the inclusion. For the mapping display, we used apple maps and google maps simultaneously to support cross platform development across iOS/Android. Apple Maps, specifically, is already built into the React Native library (react-native-maps), but we have enabled the Google Maps SDK for Android which requires the inclusion of an API key. On the other hand for location, we used the Expo location API for location tracking. In both of these cases, the client either directly calls the API or it interacts with the SDK (Firebase & Google Maps) which handles all REST API calls itself (to Firestore, Device Apis for location, and the rendering of the map).

What has not been implemented yet are features which require *custom* backend logic: reward/gamification systems, pattern analysis, routing & estimated time of arrival, fraud/accuracy detection (which has been mentioned in prototype plans) etc. These have been left off for future sprints pending further refinement of the application and will most likely require us to use an actual server like AWS as we perform more backend development in Python and FastAPI.

The link to our team's code repository is included here: Food Truck Tracker Github Repository

Sprint 2:

To host the app, we can use AWS for their flexible scaling, low costs, and shared responsibility model. The database layer will use either Postgres or SQL to store data about food trucks, as well as handling any queries efficiently. Backend services will be developed in Python + FastAPI for quick iteration. However, we will leave room to expand into Node.js if specific libraries or integrations require them. The client layer will consist of iOS, Android, and Web applications that interact with this backend layer via RESTful APIs.

Data Sources: The prototype will require user input, location data, external APIs, and user authentication.

- User input: Customers and vendors will provide updates like location check-ins, reviews, food images, or timing updates.
- Location Data: Google Maps API will help provide location baselines and routing, as well as metrics for user proximity to food trucks.
- External APIs: To pull location data.
- Authentication: Stores user information (favorite restaurants, food preferences, dietary restrictions, city location) to personalize their in-app experience.

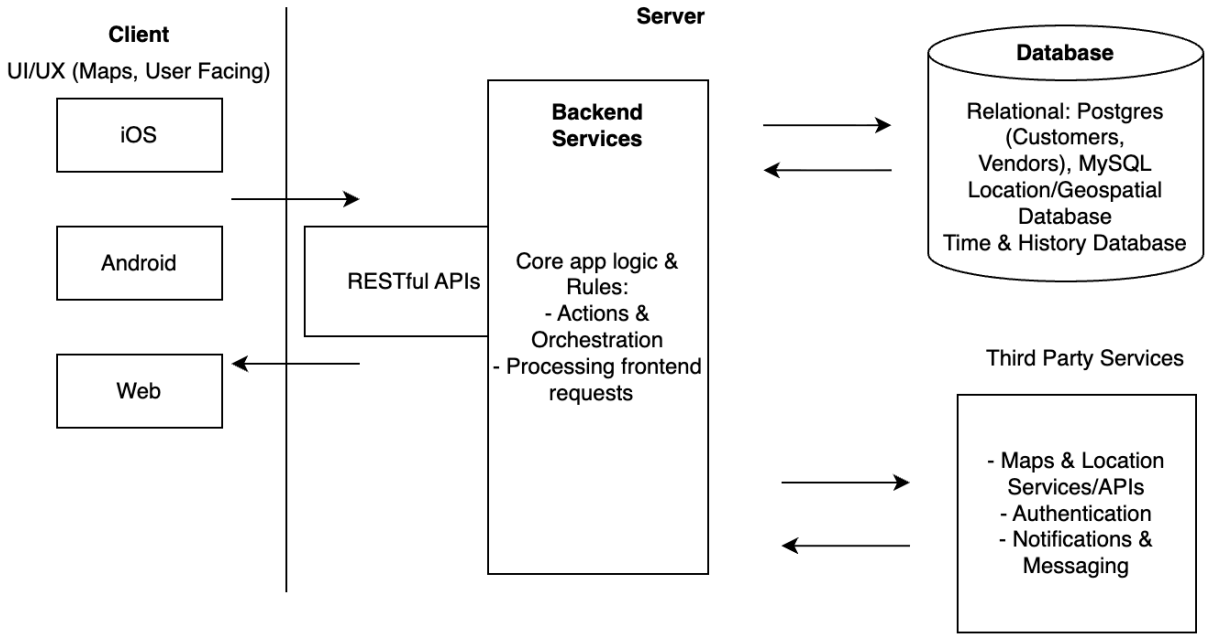
Data Flow: The data flow begins at the client (user-facing app), routes the requests through the backend logic, and then integrates with the databases and third-party APIs.

Features:

- Frontend (client layer): map-based UI/UX, photo uploads per truck, real-time location updates, vendor dashboards
- Database: user/vendor data, location and time history, support for geospatial queries (PostGIS)
- Backend: rules, validation, routing requests
- Third-party APIs: Google Maps, Firebase authentication, push notifications

Initial Architecture:

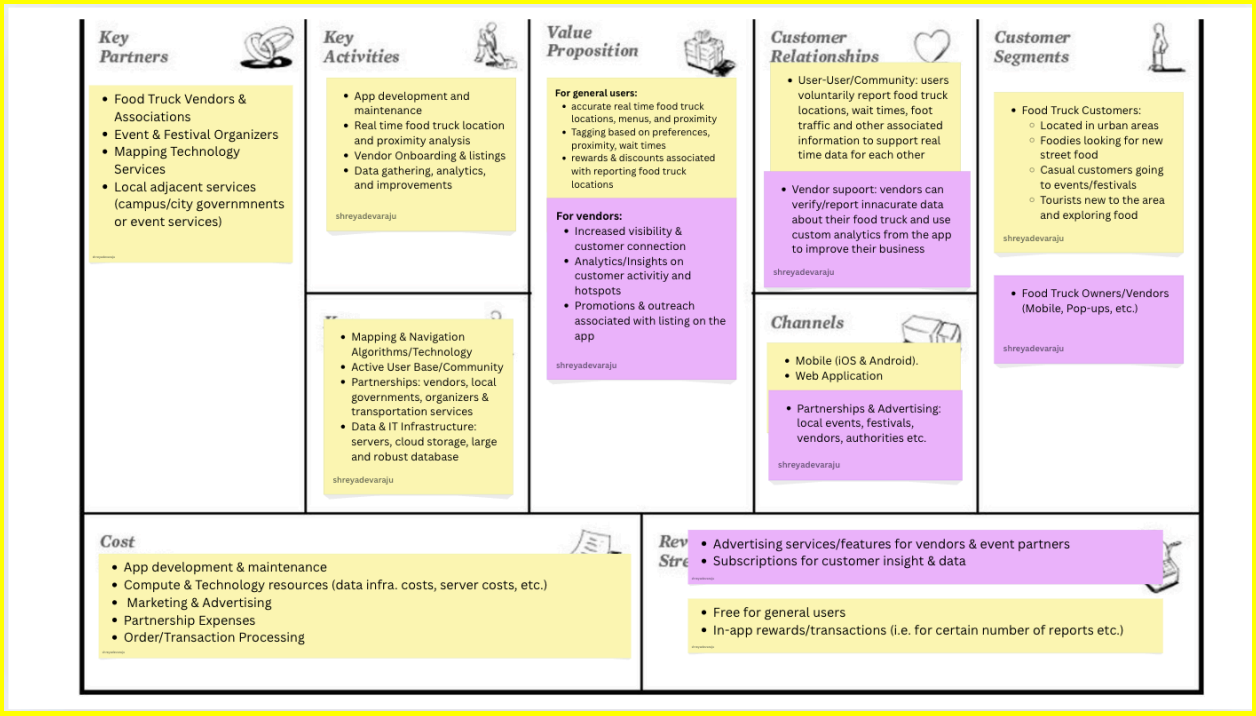
1. Client Layer (iOS, Android, Web) for user-facing maps and app interaction
2. Backend Layer (FastAPI, Node, or Java Spring Boot) for logic
3. Database and Services Layer consisting of Postgres/SQL database for information storage and integration with external APIs (Google Maps, Firebase)


Client
UI/UX (Maps, User Facing)
iOS
Android
Web
Server
Backend Services
RESTful APIs
Core app logic & Rules:
- Actions & Orchestration
- Processing frontend requests
Database
Relational: Postgres (Customers, Vendors), MySQL Location/Geospatial Database
Time & History Database
Third Party Services
- Maps & Location Services/APIs
- Authentication
- Notifications & Messaging

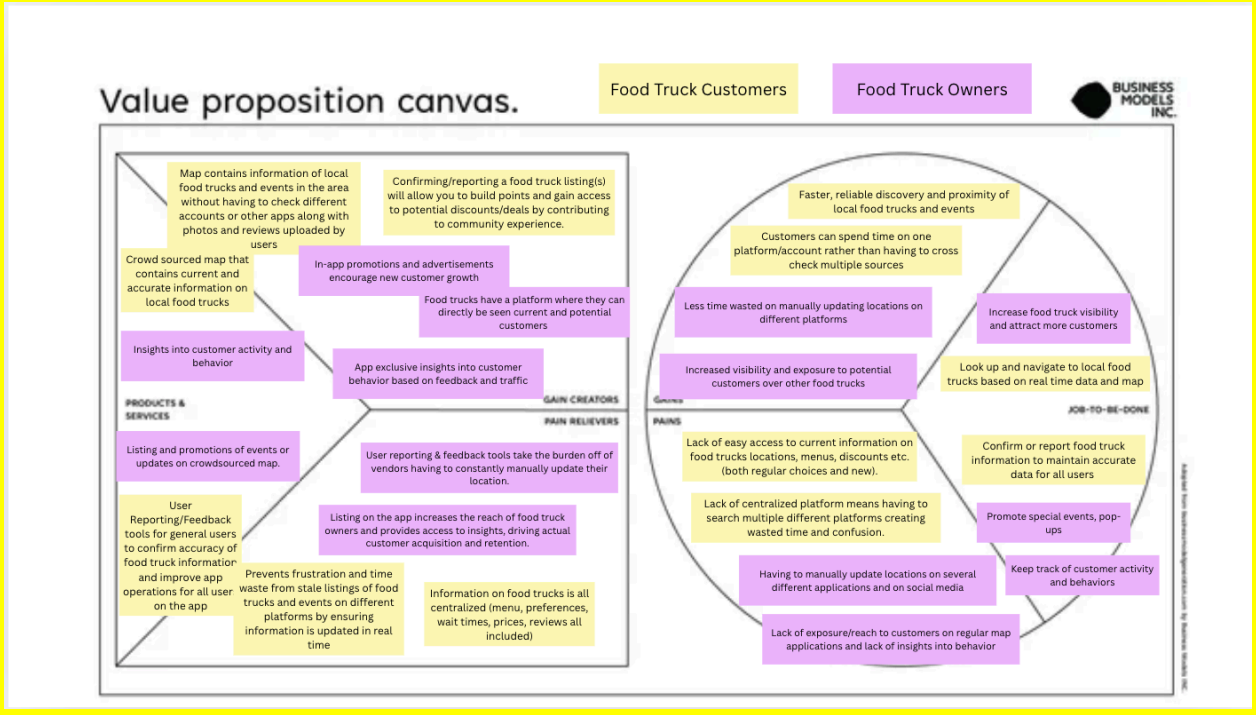
Value Proposition Canvas

Sprint 5:

Business Model Canvas (BMC)



Value Proposition Canvas (VPC)

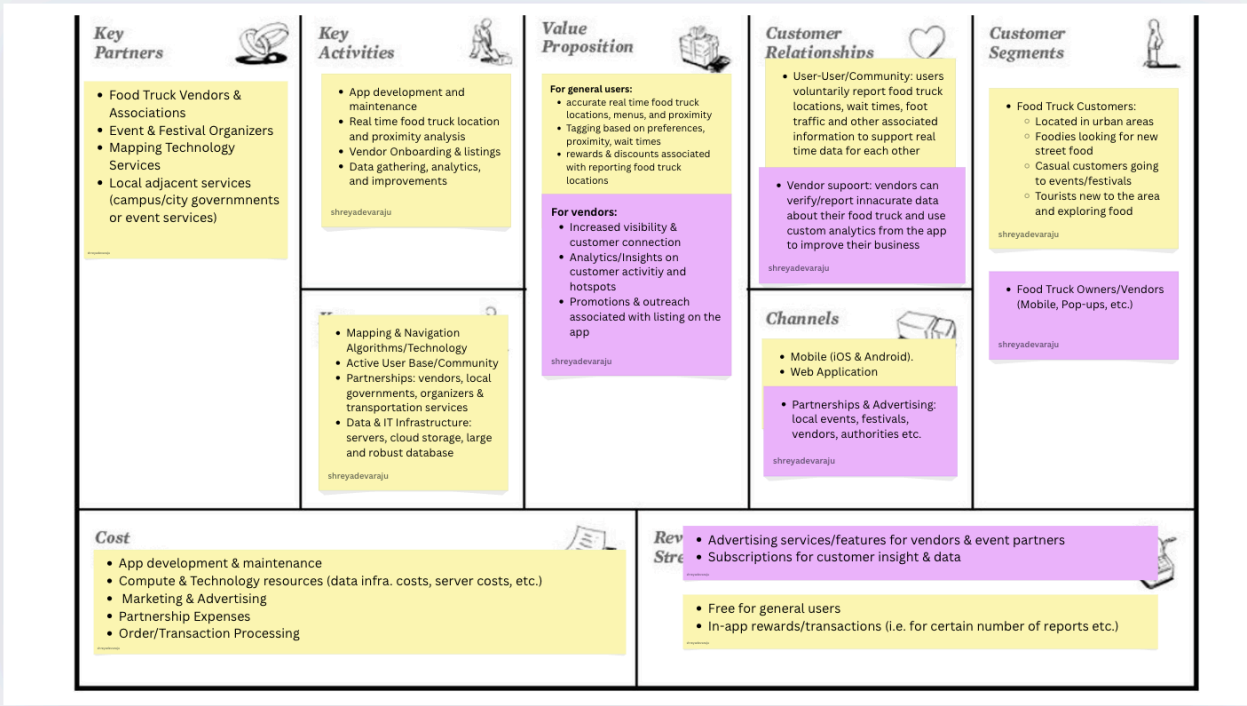


In Sprint 4, we mentioned a few key features that had been left off for future development: particularly discounts and gamification, as well as insights on customer behavior. Although we initially planned on addressing gamification in Sprint 5, we shifted to focusing on our other key customer segment, vendors, since our past testing has so far been solely focused on food truck customers (see Learning Prototype Plans for Sprint 5). As such, gamification as well as testing on rewards and discounts has been left for future development and will most likely be more relevant when we acquire a larger user base. Our team did however, get to address advertising: our current learning prototype features and tests a discover page where select vendors can be featured and as such be promoted on the app with more priority than vendors who are not featured. Although we developed a dashboard to fulfill the requirement of a second platform and to help us collect key metrics for both customer segments, as we continue testing and preparing for launch, we feel it's important to clarify that this dashboard is NOT for vendor usage but instead for admin usage. As such, this current prototype *does not yet* address insights into customer behavior which, like gamification, has been left off for future development. This follows the same reasoning as last time, being that such a feature most likely won't draw attention from vendors without first having a large, established user base – a stage which our platform has not reached yet, and because testing for this sprint was mainly focused on vendor verification.

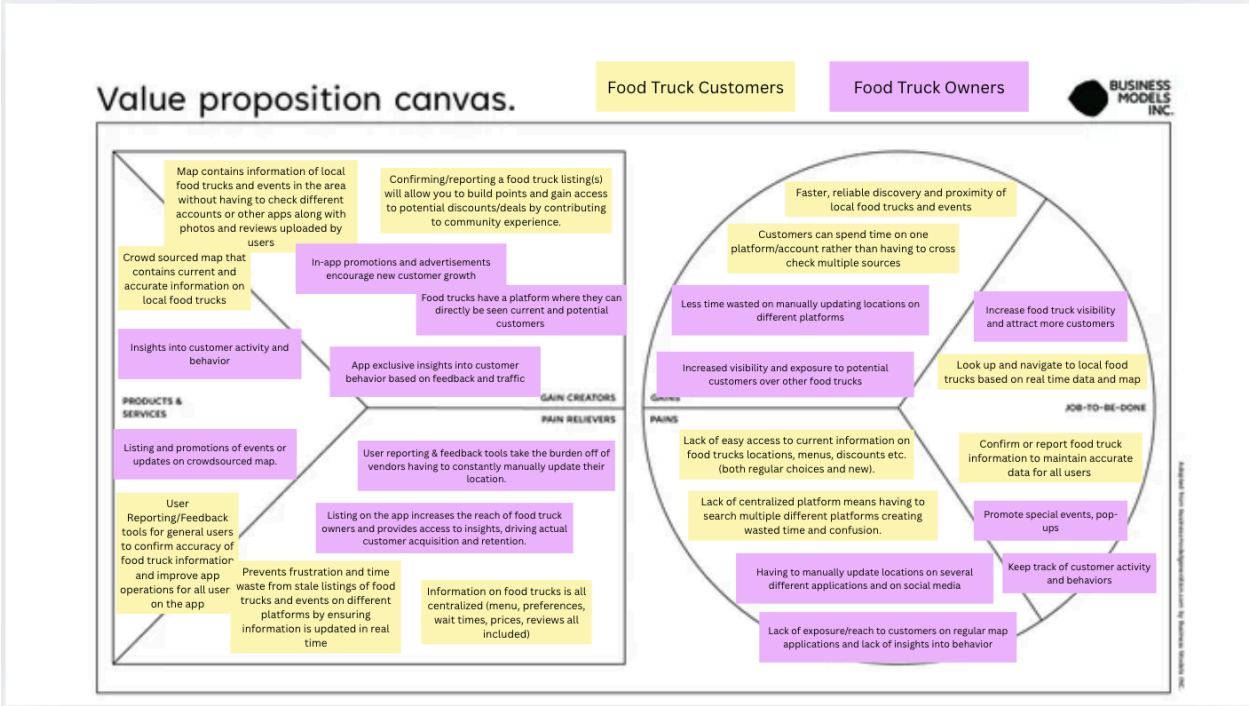
Again, our VPC and BMC were largely complete to begin with. Furthermore, no significant pivots were made from the last sprint in terms of customer segments, and we have not discovered or added additional features in our interviews/testing that weren't already mentioned, so the status of the current BMC and VPC is unchanged for the last sprint.

Sprint 4:

Business Model Canvas (BMC)



Value Proposition Canvas (VPC)



In the last sprint, there were a number of features that we mentioned in our BMC and VPC that were left off for development during the fourth and fifth sprints: namely vendor-specific features and everything besides the basic reporting of food trucks for users. Our second learning prototype now contains a few additional features that were mentioned in previous sprints. In particular, our second learning prototype now also gives users the ability to access information about food trucks beyond location including popular menu items as well as notes about discounts/special offerings and ratings for the food truck. We have also given users the option to click and navigate to the food truck location rather than just seeing food trucks on a map.

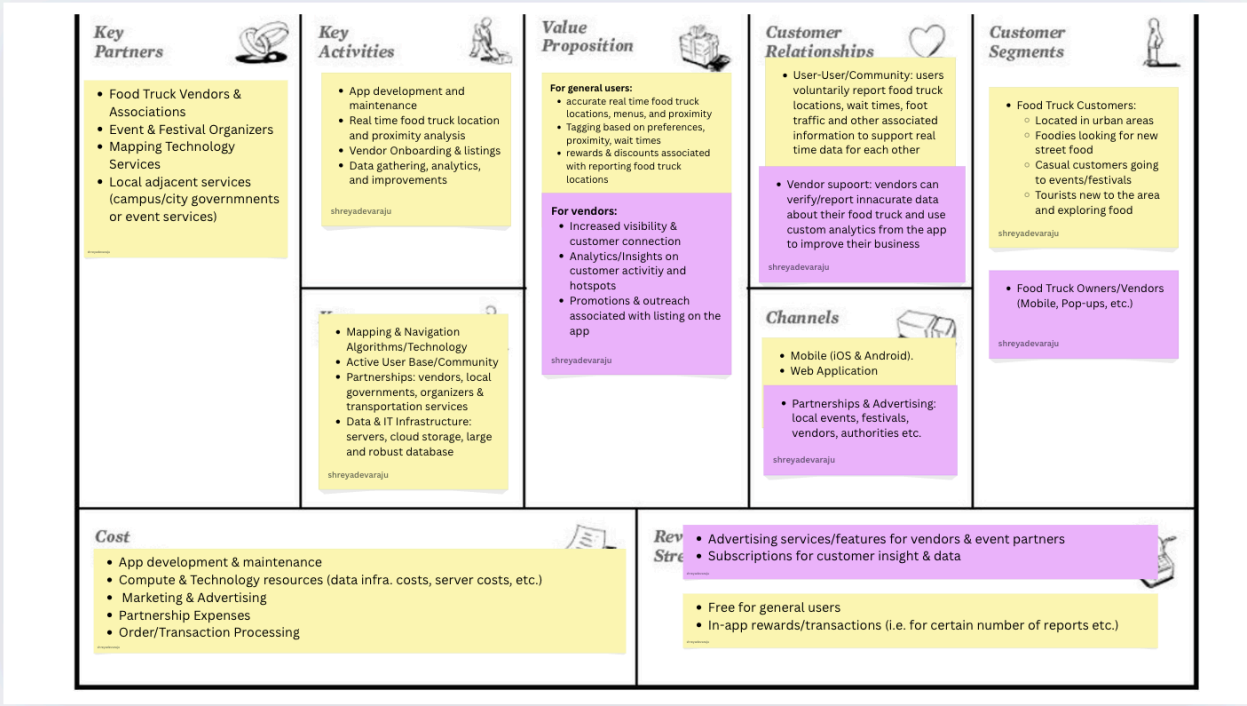
We have also begun developing vendor specific features in this sprint. Particularly, offering vendors a page to quickly check-in and report a sighting of their own food truck that's automatically verified so that, as stated in the VPC, they don't have to manually update on several different applications as well as post updates on their inventory, special notes or other information. Features that have yet to be developed include discounts/gamification as well as insights on customer behavior. While we plan to address gamification in the last sprint, we most

likely will not be integrating customer analytics into our final MVP as that feature most likely won't draw attention from vendors without first having a large, established user base – a stage which our platform has not reached yet as we're focused on testing and validation.

Beyond that, our VPC and BMC were largely complete to begin with. Furthermore, no significant pivots were made from the last sprint in terms of customer segments and we have not discovered or added additional features in our interviews/testing that weren't already mentioned so the status of the current BMC and VPC is unchanged.

Sprint 3:

<u>Business Model Canvas (BMC)</u>



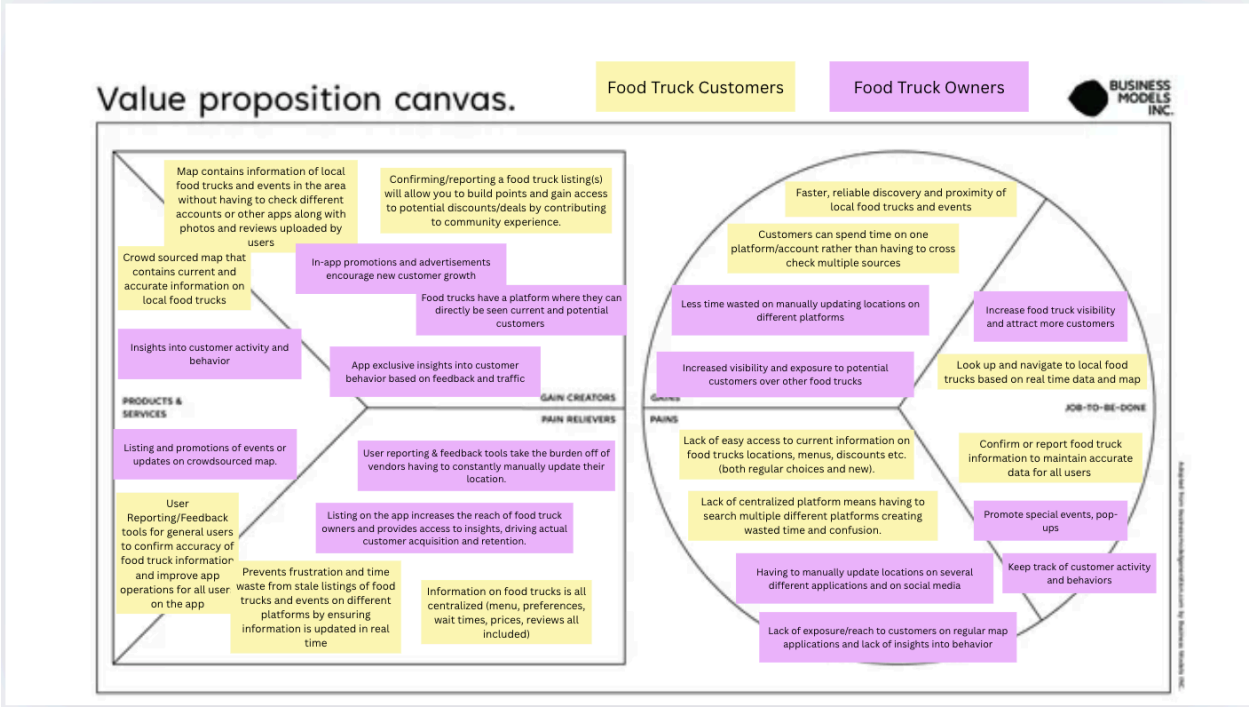
Note: The first learning prototype we developed for early testing in Sprint 3 currently primarily supports ONLY the customer reporting of food truck locations. The vendor-specific features have been left off later sprints and future MVP development.

Based on feedback in class, our updated BMC is now colorcoded by customer segment to match the VPC and also includes social media as a channel (in partnerships and advertising). As results from last sprint's online survey indicated that in-app rewards would be likely to increase adoption of this type of application, the BMC was also updated to reflect that potential revenue stream and interaction within the application (although this early learning prototype *does not*

support in-app ordering/transactions). Beyond that, no significant pivots were made so the status of the current BMC and VPC is still largely valid beyond these small changes.

Some other potential customer segments that we considered including through our domain research and class feedback were event organizers and advertisers, however, since the function goal in building these prototypes are to test the essential value for customers and food truck vendors for an MVP (who our interviews have focused on), we opted not to include event organizers or advertisers to avoid making the initial MVP too heavy and to wait on further testing and see whether the volume of user adoption justifies such an inclusion.

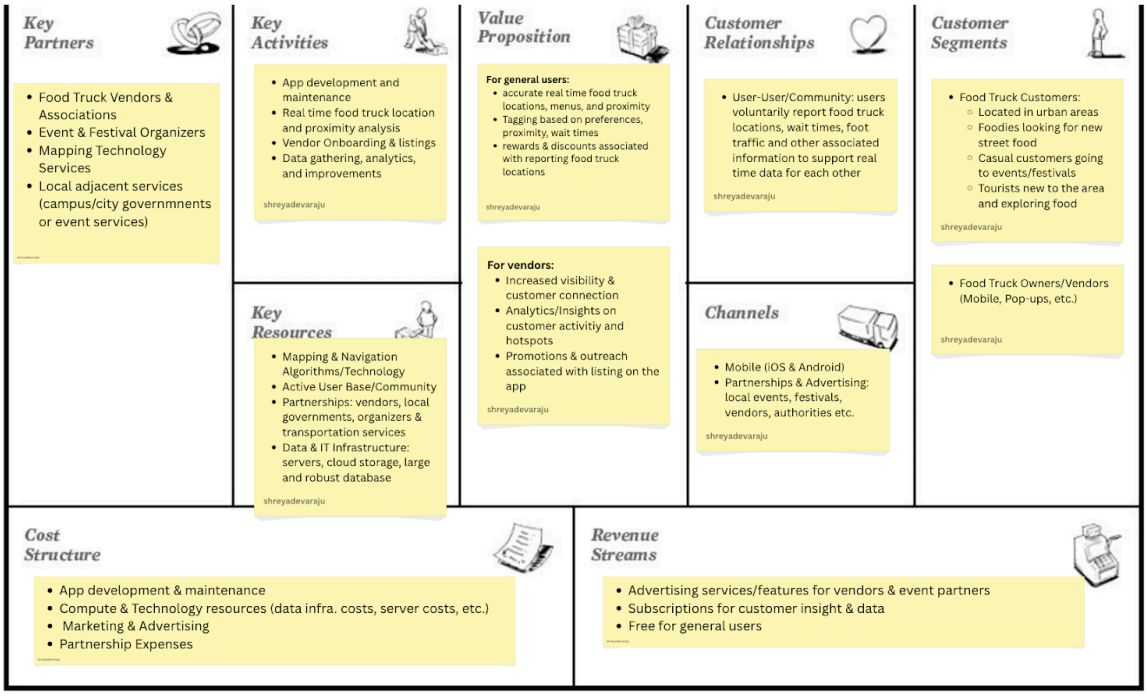
Value Proposition Canvas



Sprint 2:

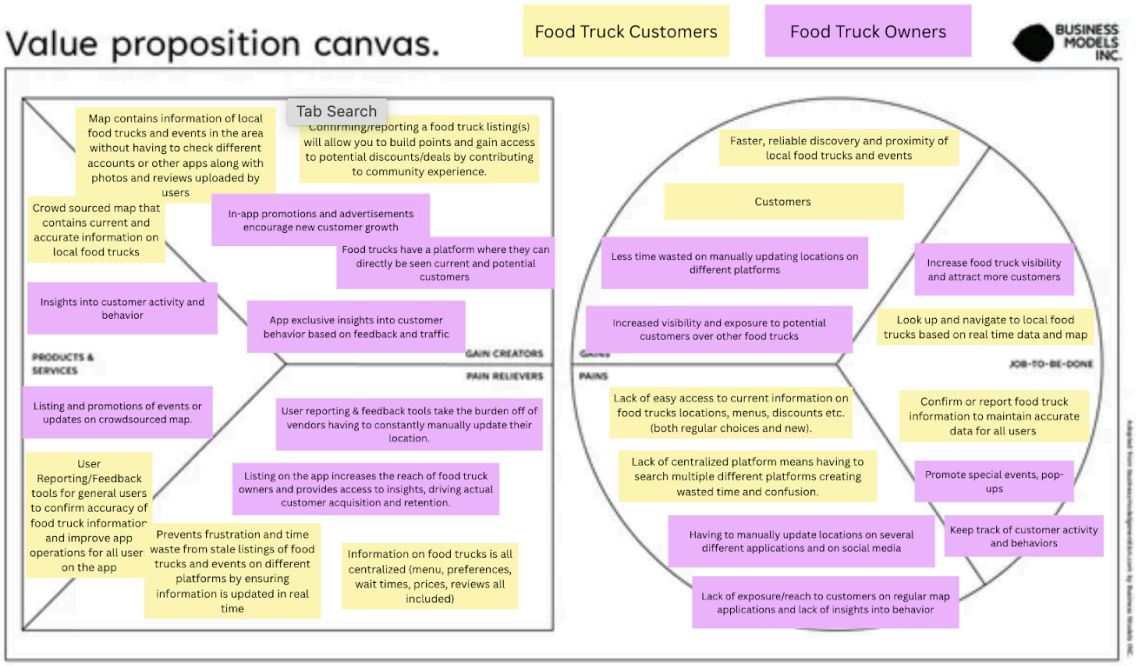
Both of these diagrams were developed with our selected approach as the basis (combination of Approach 1 and Approach 3). More information on the selected approach can be found in the Solution Approaches section.

Business Model Canvas (BMC)



Based on class feedback, we included local adjacent services (campus organizers, governments, event services) in our initial BMC to better capture potential partners for our approach beyond just customers and vendors. Mentions of customer insights and analytics were also included based on results for the survey conducted with users for Approach 1.

Value Proposition Canvas (VPC)



Note: These figures are not a representation of which features are planned to be implemented for this semester. Rather they intend to capture the overall potential of the product as a whole in terms of business impact and value proposition with customers. The actual features our team is considering to implement for the purposes of the current implementation is based on the interviews and surveys we have conducted thus far and have been noted down in the Feature Analysis

Feature Analysis

Sprint 5:

Implemented features:

1. Reporting Issues - We made it so that users can report any issues or discrepancies, by pressing on “Report issue” on the personal food truck pin page. Users can then write up their issues and press submit, where it will be saved to the reportedIssues data collections in firebase. The collection saves the text that was inputted along with what time it was submitted and what truck that it belonged to. This information will then be displayed on the personal food truck pin page.

2. Discover Page/Subscription Based Advertising for Food Trucks - The Discover page shows any food trucks that have “subscribed to a monthly payment”. This will help promote or advertise food trucks to new potential customers or users. On the discovery page each food truck has a personal card that gives the name of the food truck, its rating, cuisine type, description and some pictures. On the card for each food truck there is a “Pin/Unpin” and “View on map” action. Pinning or unpining works the same as on the map page, where it will be shown on the user profile page once pinned. View on map will send users to the map page and focus/center on that specific food truck pin if active. We came up with this feature because previous participants of user interviews/testing mentioned that they would be interested in a way to find/discover new food trucks.

   Vendor feedback: We interviewed 8 vendors. Half would pay for a monthly subscription while the other half wouldn’t. When asked how much they would be willing to pay monthly, the half that was willing ranged from $10 to $30, while the ones saying that they wouldn’t mainly said that they would only pay $5 or less, with one outlier willing to pay up to $10. The vendor mainly said that they would be willing to pay if it would help promote their business/gain profit greater than the cost, while they said that they wouldn’t want to pay if it was a waste of money/didn’t help their business. This shows that there would be interest for this feature from both vendors and users so long as the feature actually helped vendors in promoting their business by gaining more new customers.

3. Vendor Verification (A/B testing) - For Vendor verification we came up with two ways vendors can get verified: vendors must submit photo evidence (for example a social

media profile) to be approved, and or vendors/food trucks will be verified after enough users have confirmed their food truck.

We also made two different versions of the app where vendors can still see reminders and use the app even if they haven't been verified, while in the other version, vendors can't access the report, profile or discover pages if they haven't been verified.

Vendor feedback: We interviewed 8 vendors. A majority (5/8) felt that there was a need for vendor verification using photo evidence but some had concerns about how these photos would be verified if for example someone were to submit a fake photo. Many vendors express that they liked that they could use, see and get an understanding of the app beforehand without having to go through the hassle of having to get verified first, so that they know if they'd want to use it in the first place.

4. Dashboard Analytics: Users that are admins can access or see the dashboard page, where analytics will be given such as total users, average pinned truck per person, total reports, average reports per reporting user, average, issues reported, average issues per truck, top trucks that week, average reports a day per vendor, top vendor that week, and average rating per truck.

Nonimplemented features:

Gamification - These are features that would encourage users to use the app such as streaks. This could be a streak for number of confirmations or new trucks gone in a given number of times. The reason we decided to not include this type of feature for now, is that we felt that this feature was unnecessary at the moment and that it would make our app feel a bit too heavy or cluttered with features, and that there were other features that are more important that we need to focus on (based on previous interviews and testing).

Sprint 4:

Features listed by order of priority based on our interviews and user surveys for the customer facing side.

1. Menu Items - Vendors can post pictures and share what is on their menus, to let customers know beforehand what is being offered.

2 (tied). Prices - On the menu the customer will be able to see what the prices are for each item, so that they can know if it is within their budgets or is worth it.

3 (tied). Wait Times - The current average wait times can be seen by customers when they click on the food trucks, and are calculated based on user data.

4. Current Locations and Routes - This allows customers to see how far away the food trucks are, how long it would take to get there, and directions for how to get there similar to google maps or waze.

5 (tied). Rating/Reviews - Customers can leave reviews and ratings for other customers and the vendor to see after having eaten from the trucks.

6 (tied). Discounts/Deals - Vendors can show any special discounts or deals that they are having on their profile which customers can see after clicking on the truck icon on the map.

7: Festivals and Events - Customers can see and get notified of special events or festivals happening as well as what is available at the festival/event.

This tracks with what we have expected from our user demographic, which are college students and young adult workers who are interested in exploring new food options but mainly have to worry about their time and budgeting. A student or young worker would seem to take interest in a food truck if there is an item that they would want to eat/try, then they would be concerned if it was cheap enough/within their budget and how inconvenient it would be to go there and wait in line and order.

Sprint 3:
We started building our app and getting user feedback for some features that we have implemented in sprint 3 such as a user food truck reporting, a map to find reported food trucks, and user authentication.

1. Report Page - Customers can report a food truck using the report page. The app uses the users current location for the markers so that they have to be at the food truck. Then it asks for the name of the food, what type of cuisine it is, the crowd level (how busy it is), and an optional note. Then the user can submit the report, which will then have a alert pop up to show that it was successful. For testing purposes the report tells the user how many reports are needed left for that particular food truck.

   Concerns:

Some users after testing suggested maybe adding an autofill for subsequent reports. Like if it could pull from existing reports or vendor profiles so that customers only need to input the name.

2. Markers/Pins - After a food truck is reported it is given a pin on the map. For testing purposes we made it so that for the first 2 reports the pin will show up as gray (pending) and once it reaches a threshold of 3 reports it will switch to green. After pressing on a pin or marker, it will show the name, cuisine type, and crowd level for the food truck and say whether it's been verified or not.

   Concerns:
   Some feedback that we got is that unverified pins or markers shouldn't appear on the map. Also that maybe vendors should be the ones verifying the reports instead of customers. Another feedback that we got is that all the pins are green for verification and maybe should be color coded based on how busy the lines/crowds are.

3. User Authentication - When opening the app users are taken to a login screen where they can login as a customer or vendor. If they don't have an account then they can sign up by giving their email and password which they then can use again to log into the app.

   Concerns - The login it page maybe should remember login in information for convenience and potentially use two step authentication for security

Learning Prototype Results

Sprint 5:

Data: Sprint 5 Testing Data (With Vendors)

During Sprint 5, we used the most recent iteration of our application as a learning prototype to validate the vendor verification process, test the Discover Page as an advertising model, and analyze user behavior using the reporting issues system. This sprint used user testing, A/B experimentation, and the analysis of quantifiable in-app data collected through the Dashboard. Our goal was to understand how vendors, in particular, interacted with these core features.

Vendor Verification A/B Test

To assess how different onboarding approaches influence vendor engagement, the team developed and tested two verification flows with eight vendors (four in each version):
Version A: Vendors were required to complete verification before accessing the rest of the app (this can be done by submitting social media photo evidence). Vendors that weren't verified are unable to access the report, profile or discovery pages.
Version B: Vendors were allowed to freely explore the whole app and verify at any time through persistent prompts.

Findings

- Comfort exploring the app:
  Vendors in Condition B reported significantly higher comfort (avg. 4.75/5) compared to Condition A (avg. 2.5/5). Allowing exploration helped vendors understand the purpose of verification more naturally.

- Perceived professionalism/trust:
  Version A had higher trust ratings (avg. 4.25/5 vs. 3.5/5), suggesting that having a stricter verification process feels more trustworthy to users/vendors.

- Likelihood of completing verification:
  Version A was slightly less likely to complete the verification process then version B (3.5/5 vs 4/5). This aligns with what we expected as a stricter verification process increases overhead for vendors, indicating that maybe a solution that isn't too complicated would work better.

- Frustration (Condition A):
  Vendors expressed moderate frustration (avg. 2.5/5), showing that not letting vendors fully experience the app without verifying is frustrating,

Insights

A majority of vendors felt that there was a need for vendor verification using photo evidence and the data supports this showing that the app felt more professional and trustworthy. Some vendors had concerns about how these photos would be verified if for example someone were to submit a fake photo. Many vendors express that they liked that they could use, see and get an understanding of the app beforehand without having to go through the hassle of having to get verified first, so that they know if they'd want to use it in the first place. This is supported by the data as vendors who couldn't experience the app fully without verification were more frustrated while vendors that could were more comfortable and likely to go through with verification.

Discover Page and Subscription-Based Advertising

Sprint 5 also focused on evaluating the new Discover Page feature, which allows vendors to pay a monthly subscription in order to be promoted or advertised to users. Our team presented this concept to vendors during testing to assess perceived value, willingness to pay, and concerns. Note that we did not test actual subscription models (which would require more in depth A/B testing) as this would have overloaded testing and made it tough to pinpoint what affected vendor sentiment: as per both class and TA feedback we decided to keep testing focused and narrow to these two questions.

Testing Results

- Half of the 8 vendors indicated they would be willing to pay a monthly subscription in order to be promoted or advertised.
- The half that was willing to pay, ranged from $10 to $30 in terms, with the average being $18.75 in what they were willing to pay a month
- The half that weren't willing to pay said that they would only pay $5 or less, with one outlier willing to pay up to $10. The average for this group was $6.25
- The vendor mainly said that they would be willing to pay if it would help promote their business/gain profit greater than the cost.
- The main reasons vendors gave for why they wouldn't pay was that they felt it would be a waste of money if it didn't actually help their business/increase customers and profits.

Insights

The results showed that there is a decent percentage or demographic of vendors that are willing to pay a monthly subscription so long as it actually helps to promote their business, and increase their profits. Indicating that finding the write amount to charge and making it so users actually use the discovery feature and go out to try new food trucks is crucial for this feature to work.

Reporting System and Verification Logic

With the reporting and verification system fully operational, Sprint 5 provided the first opportunity to test the complete user flow. Users can now submit sightings, confirm existing

reports, report any issues or discrepancies and interact with the automatic verification logic. This makes the app more credible and trustworthy for both users and vendors.

Dashboard Analytics as a Research Tool

The Dashboard introduced in this sprint served as an internal tool to analyze real user and vendor activity. Our team examined metrics such as number of reports, number of pins, average sightings per user, and trends over the last 7 days. These analytics helped us understand: how often users pin trucks, how frequently vendors check in, which trucks are most active during the week, how many confirmations typical reports receive, and patterns in user reported data across time. The dashboard confirmed that user-generated data followed consistent patterns and that the verification system operated as intended. This quantitative layer added a new dimension to our understanding of user behavior and will guide future improvements in trust, transparency, and vendor engagement.

Sprint 4:

During Sprint 4, our goal was to allow users to interact with real-time map data, reporting, and verification features. For this sprint we decided to test two different versions of the app. The main difference between the versions was the personal pin page for the food trucks whenever users would press on a pin. The testing of this sprint focused on usability and UI comprehension, namely verification indicators and data transparency cues ("verified by x users," or "last confirmed x minutes ago") would affect user trust and engagement with the food truck pins.

Development and Implementation

- Implemented separate profile pages for users and vendors
    - For users: favorite trucks and phone number (optional) for push notifications that can be enabled per-truck
    - For vendors: menu and prices (this is where vendors can edit their information)
- Separate sighting/reporting truck page for vendors
    - They only need to input their location, and optionally, how much food is left since the rest of the information (like truck name, type of cuisine, menu, and prices) stays relatively the same over time (looks different for users) (done)
- Favorites (Pinning System):
    - Users could pin and unpin trucks to save favorites.
    - They can see pinned trucks on their profile page

Methodology for Testing

For version A we distributed the prototype with a QR code to users that they can then open the app using the Expo Go. This worked for both iOS and Android. We believed this would allow us to gather more data/participants. Each participant received:

- Login credentials (Firebase test accounts)
- Instructions to report, confirm, and pin food trucks
- A google form to answer usability and quantitative questions: https://forms.gle/y7WKJxko5LUYdnf28

But many participants experienced issues with using the Expo Go app so we just decided to conduct all tests in person, to help set up the app, observe real-time app usage, and focus on whether users understood verification and how they responded to data appearing in real time.

For Version B, we conducted all user testing in person. With feedback that was manually recorded. We mainly asked the same questions for the usability testing for versions A and B so that they could be directly compared.

User Feedback and Observations

After testing this is the data for the following versions:

- Link to Version A Data: Untitled form (Responses)
- Link to Version B Data: Sprint 4 Version B User Testing Data
- Participants: 14 users total

Version A Results:

- Users appreciated the clean layout, “popular items” section, and crowd level color cues.
- Common requests included photos, hours of operation, pricing/expensive level indicators and distance to truck display.
- Some found the modal “too busy” or “confusing in layout,” suggesting simplified UI flow.
- Quantitatively, average satisfaction = 3.6/5 and recommendation = 6.5/10.

Users were neutral about trust, often commenting that photo evidence or more confirmations would increase confidence. They wanted “proof of authenticity” such as timestamps or verified badges. Average report latency was 2-3 seconds after submission.

Version B:

- 4/5 users noticed and understood the verification banner.
- Average confidence in truck accuracy 4/5

- 80 % said the verification banner was “somewhat” or “very helpful.”
- Average decision time reduced by around 18 % (compared to Version A), with fewer abandonments and more modal interactions.
- Users who navigated to a truck mentioned that seeing “verified by x users” made them more confident.
- However, some still felt uncertain when confirmation counts were low (< 3).
- Several requested that the map auto-zoom on their location to reduce confusion and make nearby trucks stand out.

Key Findings & User Insights

- Verification status directly increased user trust and interaction rates.
- Users preferred around 4 confirmations as the threshold for “verified.”
- Most desired photo proof of location for user trust.
- Several suggested in-app confirmation prompts (“Is this truck still here?”) to encourage user participation.
- Visual simplicity was valued: users wanted data transparency without information overload.

Sprint 3

For this sprint, we created the MVP. The goal of this sprint was to create the basic app and focus on testing functionality that users would want. Throughout code development, each user tested functionality of features as they were implemented. This was done to ensure a baseline for if they worked as they should. It also gave instant feedback for bugs. This was the primary method to catch bugs, such as the keyboard not disappearing during sign-up/log-in. After we completed all the features we intended to complete for this sprint, we moved on to user testing.

As mentioned previously and as suggested in class by peers, we mocked vendor testing to remain considerate of vendors’ busy schedules. Both user and vendor testing was done by classmates. Since the app was deployed through Expo Go, we were able to easily share a single build link that worked across both iOS and Android phones without requiring a full installation. The interviews were also conducted in various locations throughout Midtown Atlanta and Georgia Tech’s campus to test various pins. Instead of having users fill out a Google Form, post-testing interviews were conducted in-person to gain more through details about the app.

Link to Data: Sprint 3 Data

Key takeaways from user interview testing:

1. Many users wanted to be able to track specific trucks and get more information about their menu and prices

2. The reporting page needed to be more tailored. Some of the information for users didn't align with information vendors should see. Also, vendors shouldn't have to put in the same information about cuisine, name, and prices every time. This can be pulled from their profile. A profile page would allow them to edit this information easily.
3. Many people were nervous about not having vendor verification.

Sprint 2

Approach 1:

The Rolling Crew - Approach 1 Online Survey Results**:** Survey Results

By building basic mockups in Figma, our team was better able to visualize and understand what the application would look like from food truck customers' perspectives at a basic level. We initially included what we believed to be priority features for users based on our research conducted in Sprint 1, so we could compare it concretely with the results from the survey. Wait times, proximity, ratings, and keeping track of"favorited" trucks in particular were what we initially believed to be priority based on interviews, so in building the figma mockup we also tried to visualize what that would look like if we went with approach 1.

We obtained 18 responses to our survey. The full visualization of the data is captured in the link above while we briefly summarize the most relevant portions of our findings below.

Customers: 15/18

When initially asked how likely they would be to use a "waze" like app for food trucks, responses were divided between 2 and 4 (not very likely and likely). When asked about rewards and discounts, respondents indicated their urge to use the app would increase showing the importance of monetary incentives in active engagement with the application. When asked what information would be the most helpful to view real-time, almost all features (current location/routes, menu items, prices, wait times, ratings/reviews, discounts/deals) were indicated as important by respondents whereas keeping track of festivals and events were viewed as less important. Menu items, prices, wait times, and locations/routes received the most votes, in particular. We also asked what might keep users coming back if only a few trucks were listed initially and more respondents indicated that supporting local food trucks and rewards or incentives would keep them coming back.

Vendors: 1/18

The most important insight here was that privacy and control over location and branding seemed to be a concern from the vendor perspective and that customer analytics were viewed as an important feature. While more responses would be helpful in determining whether this is an accurate representation of general concerns, these are concerns we will need to consider as we consider potential feature implementation both for this approach or for our final chosen approach.

Summary:
Ultimately, the survey results showed that this approach needs to function beyond just reporting different food trucks to be viable for broad user adoption. For instance, on the customer side, general information on food trucks (menus, wait times, etc.) and other basic tags were confirmed to be important to users to get the most out of this sort of application. Although the majority of respondents said they rarely eat at food trucks, a broad likelihood to use the app was still there, particularly with rewards and incentives which confirmed how important the feature was and fell in line with class feedback as well. On the vendor side, privacy and brand control were concerns. However, since we only received one vendor response, we plan to follow up in interviews and confirm how much of a concern this is with more vendors. Since many of the basic tag features are already included as part of Approach 3, the basics of this approach might end up being the most valuable if combined with those from Approach 3 as well.

Approach 2:

To test Approach 2, we created a low-fidelity mock-up of a vendor-facing tool that allows food truck owners to push updates about their location, hours, and closing information to multiple social media platforms at once. The mock-up was done on paper to focus on functionality rather than design. This helped us quickly communicate our ideas of the workflow, from entering information on the app to previewing how it appears on various social media platforms. This aligned with suggestions from lecture about keeping early learning prototypes simple and focusing on concepts rather than UI.

We used the same mock-up in testing sessions with peers and vendors. We had to consult with some peers instead of solely working with vendors due to a lack of access to vendors in Midtown Atlanta. This highlights the urgency for a solution that finds food trucks nearby. In these demos, we showed how vendors can input updates and how those updates generate formatted posts across platforms. Participants were asked to provide feedback on usefulness, feasibility, adoption, and gaps from a vendor and customer perspective.

Feedback:

- Vendors liked the idea of saving time by only having to post once. They also confirmed that during busy hours, it is difficult for them to post to multiple accounts and reach a

large audience. The main concern was about whether this app would integrate into their current workflow, especially since they would still have to input the update manually.
- Customers found the idea useful for consistent updates across platforms they use. However, they noted that they would have to follow the food truck's social media in order to see the posts, which doesn't help find new food trucks.
- Similar to the feedback from customers, Canvas discussions highlighted that the impact of this solution is limited to customers that have specific food trucks in mind. They suggested moving forward with another approach that is more focused on discovery for the general public.

Summary: The paper prototype validated that a vendor tool with social media syncing could reduce posting friction, but also revealed its limits as a standalone solution. It is better suited as a supporting feature rather than the foundation of our product with the goal of discovering food trucks nearby. This insight, combined with feedback from domain research and interviews, is why we are now leaning toward combining Approaches 1 and 3 as our core direction.

Approach 3:

To test approach 3, we created two low fidelity prototypes, a paper mock up of the customer side and a paper story board for the vendor side. Both prototypes were kept very simple as suggested in lecture so that they would only convey the functionality of the different pages and the information that those pages conveyed to the vendors and customers. The mock up showed customers where food trucks were on a map relative to where they were, how far the trucks were, what they offered, for how much, etc. The storyboard showed vendors the process of setting up their profiles and how they would attract customers, as well as how to change their active status.

Feedback:
- Vendors liked how simple it was to set up their profiles and how it provided information to customers, but had concerns about having to always change their activity status. For example they forget to set the status to not operating/active and customers accidentally follow them home thinking that they are still operating
- Customers liked how easy it was to find food trucks around them and what they had to offer: menu, pricing and deals. Would like to know when vendors are closing/wrapping up so that they get there and everything is closed. Would like to have a way to discover new food trucks, maybe a recommendation of sorts

Summary: This showed us that vendors and customers generally like what this approach has to offer but would need some tweaking in terms of user experience to make usage more convenient, which is why we decided on moving forward with a combination of approach 1 and 3.

Overall Findings:

Ultimately the feedback we received on each of the prototypes for the different approaches, as well as the results from the survey, gave our team a good amount of clarity on user interest in each of the approaches and what might drive more interest. As mentioned in the above findings, based on user feedback, Approach 2, while liked for its integration-minded approach, didn't encourage as much new visibility as vendors were looking for in attracting new customers.
In contrast, the feedback in surveys for Approach 1 and the feedback to the prototype developed for Approach 3 seemed to show more promise based on user willingness because vendors were looking for ways to reach more users and customers were looking for easy, centralized access to information on food trucks and finding new food trucks.

Learning Prototype Plans

Plans for Sprint 6:

Our next learning prototype will build on the role based features we completed this sprint and move toward a more polished and scalable administrative system. Now that we have user, vendor, and admin flows working, we want to improve the experience for administrators, reduce manual steps in vendor verification, and improve the reliability of our analytics pipeline.

During this sprint, we created the vendor onboarding flow, the pending approval screen, and the admin dashboard access logic. The testing we conducted also confirmed that the system can correctly identify account roles and present the right interfaces. For the next sprint, we want to expand our prototype by focusing on two main areas: a smoother vendor approval workflow and a stronger admin backend that reduces manual work.

The most important feature in the next prototype will be a dedicated vendor review page inside the admin dashboard. Right now, administrators must approve vendor accounts through Firebase directly. This makes the process slow and creates unnecessary friction. We want to design and prototype an approval interface inside our dashboard where admins can view all pending vendors, see their uploaded proof images, and approve or deny them with one action. This will help us learn whether our data model supports admin level actions cleanly and whether the approval process feels natural when performed through our own tooling.

We also want to extend our separate admin backend so that it can handle key workflows like vendor approval, updating verification status, and storing a clean audit trail. This will help us validate whether our backend can support administrative operations at scale without relying on Firebase Console. It will also let us test how different admin actions propagate through the app in real time.

Other features we are considering, but would be classified as a lower priority, include deeper analytics charts, automatic trend detection, and push based reporting for daily vendor activity. These ideas are interesting, but they depend on having a smoother approval system and stronger backend in place first. They are not essential for answering the main questions we want to tackle in the next sprint.

The main questions we want this next prototype to answer include:

- Can administrators review vendors from within the app without going through the Firebase Console?
- Is the new approval interface intuitive, fast, and consistent for admin users? Does it get too heavy with too many tabs on the bottom?
- Does the workflow feel smoother for vendors after we update the admin review process?
- Do these changes prepare the system for more advanced analytics and feature flags later?

Plans for Sprint 5:

Developing and testing two different versions of our application with users gave us valuable insight into not just what information users would deem essential when keeping track of food trucks, but also what sort of presentation users would respond better to and what threshold of confirmations would increase trust in our application's sightings. Based on the results collected from the Sprint 3, we have decided to move forward with Version A's UI and include the information about the food truck's menu items and reviews as even those who were tested on Version B expressed a desire to see menu items and reviews before navigating to a food truck.

Beyond this and fixing any bugs with the application that came up during our user tests, the main features of the application and most of the enhancements have now been implemented.
As such, we plan to focus this next sprint on developing and testing different gamification features (most likely with A/B testing) as well as developing a stronger in-app method to verify that actual vendors are signing up and not just anyone with a verified email. Our goal in doing this is not just to increase the security and authenticity of our application, but also to determine if introducing some gamification element will further encourage consistent engagement with our application.

Key features we plan to implement include:

1. Vendor sign-up verification for security
    a. Manual verification through screenshot or some other method
2. ~~Gamification features~~
    a. ~~Streaks~~
    b. ~~Levels and Points~~

We also plan to introduce vendor testing in our next sprint to ideally prevent overfitting to customer needs, however we have largely developed the main features needed for testing with vendors. ~~Our goal during this next sprint alongside testing with users will be planning a concrete schedule for testing with vendors, determining if the experience of maintaining information on the application is quick and convenient enough to secure their continued interest, and determining whether the right level of information is being tracked.~~

As we've already focused on testing the application with customers during the past few sprints, we decided to shift our plans for Sprint 5 and solely focused on testing the application with vendors in this testing cycle since vendors are the other key user type. Focusing more on vendors was also brought up by class peers. Coupled with this feedback and in order to avoid stretching ourselves too thin in testing (and risk not conducting testing well with either group), we instead decided to focus on acquiring more vendors so that we could gain quality data and round out our MVP.

Plans for Sprint 4

Having a basic app and testing it with various users gave us perspectives that we hadn't thought about before. We were able to catch some nuanced bugs and learn about features that users would want. We finished most of the major functionality for the app. The next sprint will be about refining the app. For instance, to make reporting more tailored, users and vendors will have a separate reporting page depending on their account. Keeping in mind group discussions, peer feedback, and user interviews, we created a list of features we would like to enhance, implement, and fix for the next sprint.

Key features we want to implement:

1. Implement profile page (separate for user and vendor)
    a. For users: favorite trucks and phone number (optional) for push notifications that can be enabled per-truck
    b. For vendors: menu and prices (this is where vendors can edit their information)
2. Pin enhancement
    a. Check for the same user not reporting the same truck within a certain timeframe (fraud)
    b. Fading functionality for trucks that have not been spotted in *x* hours and removing truck functionality
3. Vendor sign-up verification for security
4. Separate sighting/reporting truck page
    a. This should be more specific for user vs. vendors.
    b. For users: they need the truck name, type of cuisine, busy level, and a warning that at least 2 other people must also report this truck
    c. For vendors: they only need to input their location, and optionally, how much food is left since the rest of the information (like truck name, type of cuisine, menu, and prices) stays relatively the same over time
5. Handling crowdsourced accuracy:
    a. The current implementation doesn't use exact coordinates and instead uses a 100 meter radius for the truck to combine results. However, for other crowdsourced information, such as light vs. moderate busy, there should be functionality to weigh the more recent one more, but also still account for older updates.

Plans for Sprint 3

For the next sprint, we plan on developing an initial base for the application with the most important features for our chosen approach so we can test those aspects of the solution with users and actually begin to gain feedback on the actual in-app experience. Due to the nature of the combined approach, our first priority is to begin development of the mapping and location tracking functionality of food trucks within the app since that is the foundation for the application and work our way through any big concerns on that feature (i.e. external geo location API concerns, data sourcing, listing concerns) in the beginning itself. We also plan to initially develop a baseline version of the user inputting feature (a potential version was shown in the learning prototype for approach 1) where users can select a location on the map and input a new food truck. Our main objectives with focusing on both of these features over others are so we can work through concerns with implementation of foundational features early on and begin showing a base level version of the app in interviews or testing later on and gather feedback on how to actually fix any initial concerns bought up with these base level features (i.e. how to address accuracy and privacy).

Although vendors are mentioned as a customer segment throughout our notebook, we don't plan to initially develop features targeted towards vendor usage during Sprint 3 because we still have some questions (concerning privacy and control as mentioned in user surveys and feedback) that we'd like to gather further information on from vendors themselves and discuss as a team. By the time we get to Sprint 4, our goal will be to use the feedback we've gathered on the initial base features for the application and remaining concerns to move into creating an end-to-end developed app with the rest of the features that can be tested as a whole and that starts to move us further towards refining our approach so that it reflects an MVP that can stand on its own in the market against competitors.

Plans for Sprint 2

The purpose of our sprint 2 learning prototype is to determine whether the app should be primarily vendor-facing or both customer-facing and vendor-facing. To that effect, we plan on conducting surveys with both of these user-groups and narrowing down on a quantitative level the exact level of need for this type of product and the types of features they prioritize. This way, we can also effectively narrow down what features should be prioritized for an MVP version of each approach and finalize what minimum requirements should be set for our product based on level of need and whether or not the features required are in-scope during the semester. This way, by the start of Sprint 3, we can begin to plan out how to create hi-fidelity mock-ups and prototypes based on our research. Based on our initial interviews, our team has outlined a few features that we believe will be most important to addressing the issue as they've shown up as common complaints. We've outlined them in order of importance to reflect what we believe is

the biggest priority and what we believe may not be the biggest priority: conducting surveys will allow us to validate or change this list based on actual data.

Features:

1. **Live tracking & Proximity**
2. **Centralized app for vendors**
3. **Automatic Updates (Social Media & Users)**
4. Wait times, hours, ratings/reviews
5. Favorited Trucks & Suggestions

Code Review

The Food Truck Tracker application is organized into modular React Native screens that coordinate through React Navigation, Firebase Authentication, and Firestore. At the top level, App.js acts as the central orchestrator: it sets up the navigation container, defines a bottom tab navigator, wraps the Map tab in a nested stack navigator, and only shows an admin only Dashboard tab when the signed in user's Firestore document has isAdmin set to true. It also listens for authentication state changes and passes the isAdmin flag down into the tab navigator, which keeps the codebase cleanly separated between navigation concerns and feature logic while still allowing shared behaviors such as error boundaries, loading indicators, and authentication based UI. These navigators allow different parts of the UI, such as the Map, Report, Discover, Profile, and Dashboard screens/pages to communicate with one another while keeping a clear separation of responsibilities. For example, the Map tab contains its own stack, enabling screens to receive parameters such as focusTruckName, which controls map centering when navigating from other screens or pages.

MapScreen.js:

Each major feature of the app is implemented in its own screen file. The MapScreen handles real time map rendering by listening to or pulling from Firestore's sightings collection. Pins are created or generated from the sightings collections documents using logic to filter out invalid coordinates and group duplicate sightings by truck and location so that each truck is shown as a single pin. Pressing on the food trucks pins opens a small callout sheet with a "Pin/Unpin" action that writes to the favorites/{userId} collections document in the firebase database using arrayUnion / arrayRemove, and those favorites are later displayed on the Profile screen. A more detailed personal food truck sheet also pops up after pressing on the pins that displays crowd level, description, verification state, recent confirmation time, issues reported (including what and when reported), ratings, distance from user, notable dishes and options to report issues or navigate to the food truck. On this personal food truck pin page, pressing on "Report issue" will open up another UI box where users can report an issue and press "Submit", which will write to the reportedIssues document collection in the firebase database, which will then be displayed on the personal food truck pin page along with when the report was sent. On the personal food truck pin page, pressing on the "Navigate" button opens Apple or Google Maps with the truck's coordinates, tying together live Firestore data, React Native components, and native mapping intents in a single interaction.

ProfileScreen.js:

The ProfileScreen displays user specific backend data for different parts of the interface. ProfileScreen.js reads favorites/{userId} in real time to display "Your Pinned Trucks" as a list, allows users to unpin trucks, and exposes notification controls using helper functions in notifications.js to subscribe or unsubscribe from truck specific push notifications. The Profile

screen interacts closely with the Map screen through navigation. When the user selects “See on map” for a pinned truck, the Profile screen navigates to the Map page and passes a parameter (focusTruckName), which MapScreen detects in a useEffect. This triggers the map to go to the correct food truck pin. This interaction demonstrates how separate screens communicate through shared navigation parameters and how the logic such as navigating to a truck is kept within the Map component to ensure consistency.

DiscoverScreen.js:

The DiscoverScreen, which supports a subscription based promotion or advertisement listings, demonstrates another layer of interaction. This screen reads from the advertisedTrucks Firestore collection, where each document contains fields such as foodTruckName, cuisineType, description, imageUrl, and isActive (meaning is verified), and renders each active document as a card with the truck’s name, tagline, cuisine, a placeholder star rating, and two main actions: “Pin/Unpin” and “View on map”. Advertised trucks share the same pinning system as sightings: pressing “Pin” or “Unpin” modifies the same favorites collection, and pressing “View on Map” uses the same navigation flow to center on the food truck’s pin. This gives the Discover page a consistent user experience despite being powered by a different Firestore data source. Adding this page required adjustments to Firestore security rules (separating public reads from admin-only writes), which improved the overall safety and scalability of backend interactions.

ReportScreen.js:

The ReportScreen is the primary data entry interface for the food trucks and acts as a bridge between users, vendors, and the sightings collections data in firebase. When users go to the report sightings page, ReportScreen.js requests foreground location permission via expo location, reverse geocodes the user’s coordinates into the address, and displays that address in a “Current Location” banner with a “Refresh” button. It then loads identity information from AsyncStorage to determine whether the current person is a regular user or a vendor. For vendors, it attempts to load the vendor's document (first by user id, then by email) so that the form can be prefilled with the food truck’s name and cuisine type and a yellow “Checking in as …” banner that can show which truck is currently being checked in. The UI changes based on this role: regular users see required fields for truck name, cuisine type, and crowd level and optional fields like notes and notable dishes, while vendors see an additional inventory level selector.

The reporting logic combines this UI with backend behavior. Before saving, ReportScreen calls findSimilarSightings, which queries sightings for recent documents matching the same truck name and filters them by time (last hour) and distance (within 100 meters using a Haversine distance helper). It then uses reporter IDs/emails to count unique reporters, not just total reports. If the truck already has a verified sighting, the UI informs the user and redirects to the map instead of creating unnecessary duplicates. Otherwise, it adds a new sighting document containing location, crowd level, optional inventory state, additional notes, and a list of “Popular

Items" that the UI saves. For vendors, the sighting is immediately marked status: "verified" and treated as a check-in; for regular users, the sighting starts as pending. When three or more unique reporters have contributed within the recent window, verifySighting updates the relevant documents to status: "verified" and sets verifiedAt, closing the feedback loop between crowd reporting and map verification.

DashboardScreen.js:

The DashboardScreen provides an admin's only "analytics" interface that uses data across the collections. It is only reachable by users who are authenticated and have the isAdmin flag set to true. The dashboard page counts all of the documents in the users collection, and uses the computeMetrics function that fetches users, favorites, sightings, and vendors. From these snapshots, the dashboard computes metrics such as total users, average pinned trucks per person (sum of favorites divided by total users), total reports, average reports per reporting user (using unique reporter IDs), averages "issues reported" per sighting using confirmation counts, average issues per truck, top trucks this week based on sightings in the last seven days, average reports per day per vendor, and average ratings where vendor documents contain rating fields. The results are rendered in a card as labeled text metrics, giving admins an overview of how users and vendors are engaging with the platform. The screen also includes explicit error handling for permission failures, displaying a simple explanation and even suggesting a Firestore rule snippet when reads to the users collection are blocked.

Underneath the visual UI, the app uses helper modules for shared functionality. firebaseConfig.js initializes Firebase and Firestore, while notifications.js manages Expo push token registration and subscription logic. These modules allow screens such as Profile or Dashboard to access shared functionality without duplicating code. Firestore listeners (onSnapshot) are used throughout the app to achieve real time updates. For example, the Profile screen monitors the favorites collection to immediately reflect pinned or unpinned trucks, the Map screen monitors sightings for live location updates, and the Discover screen listens to advertisedTrucks to show or hide promoted trucks without requiring a refresh.

Some of the more challenging implementation details involved integrating Firestore's real time database with React Navigation's nested routing. For instance, when "See on map" initially failed to center/focus on the selected truck, the issue was traced to navigating only to the tab and not the inner map stack. Fixing this required navigating using the structure navigation.navigate('Map', { screen: 'Map', params: {...} }). Another challenge involved Firestore security rules, especially when newly introduced collections like advertisedTrucks defaulted to restricted access. Resolving permissions required careful rule design, including adding helper

functions like isSignedIn() and isAdmin(), ensuring the catch all rule occurred last, and explicitly permitting read access for advertised trucks while keeping write privileges restricted to admins.

Overall, the interaction of these components, the real time Firestore listeners, navigation parameters, shared helper modules, and modular React Native screen components, creates an interconnected UI. Each screen remains responsible for its own presentation and logic, while a combination of shared state, Firestore data, and navigation props ensures that user actions in one feature (pinning, viewing, subscribing) update the rest of the app immediately and consistently.

Future Direction

Throughout the semester, as a team, we had the chance to learn and gain practical experience with several different aspects relevant to the development and validation of our application. Conducting customer interviews, testing high-level prototypes, slowly narrowing down and discovering new features to add value to a mobile product as we moved through each user-testing phase, all allowed us to build this understanding at a rapid pace.

Based on our progress so far and the questions we have left, if we were to continue this project in the future, we would most likely plan on continuing to test our application on a somewhat larger scale than before, particularly focusing on conducting more testing with vendors. In our past sprints, we were able to test with smaller sets of users (typically 10-15), which worked well for early validation, while also not being so small that our results were no longer reflective of general users. However, since we would want to release the application with a set of early adopters, instead of testing with 10-15 users, for instance, we would hope to release the application to more users in the future and gather more data – ideally 30-50 – and gradually test on a larger scale. We would also like to test for longer periods of time (a few weeks at a time), so that we can better understand what effect certain features have on encouraging consistent engagement (gamification and notifications, for instance).

Furthermore, besides Sprint 5, we conducted past user testing rounds with general users only, in part because we struggled to schedule tests with vendors, but mainly because our focus initially was on clarifying key questions about the main features of our application from the perspective of the general user (without buy-in from general users, it wouldn't matter if the application was incredibly well developed for vendors). As such, in future sprints, we would like to focus more heavily on vendor testing and acquisition, such that as we begin to scale out to testing with larger groups of users, food truck vendors are also more motivated to use the application and gain more exposure.

In terms of actual implementation, while this isn't an immediate concern, our team would also start to have more discussions on the development of a custom backend that can function beyond what Firebase allows, especially as we try to further expand on what we have currently. Also, as we've mentioned in the learning prototype plans, we would move to polish our administrative system and scale better beyond where it's at now, so that we can better monitor activities on the app and analyze user behaviors. For instance, if we tested with users and vendors simultaneously, we could better track their behaviors in a more built-out dashboard and use collected metrics for future development. Although this is not an immediate concern for the future, eventually, we would like to move towards actually having vendors view customer insights (as mentioned in the BMC and VPC), so as we work towards gaining more users and further building out the application, we would shift towards exploring what this would look like in practice as well.

Overall, we feel that we've ended our run in this course with a relatively strong version of our solution approach that's shown promise and can be viewed as a minimum viable product. As we consider the future, if we continue the application, we would hope to build on this progress and gradually test with larger groups of users at a time to get prepared for the possibility of actually launching the application at a larger scale.

Concerns

Sprint 1: Our main concern is feasibility. Food trucks have a smaller customer base, which makes it difficult to find enough users and vendors to test our prototypes and solutions. This narrows down the interview pool that we can gather and collect insights from. Moreover, food truck vendors are typically very busy, so being able to interview them and get them to test our product may be difficult. However, being in a major city, compared to a suburban area, should improve our chances of securing vendors and customers for our study.

Sprint 2: We combined our approaches from Sprint 1 into one product to move forward with as our proposed solution. However, with so many features, the app seems dense. One concern we have is the feasibility of implementing all features. We are also concerned with whether vendors will actually follow through on using our app for testing. Although we can convince students and friends to use it, a possible problem can arise with vendor adoption. Moving forward, we should look into possible vendors beforehand, rather than first developing the app and then searching for vendors.

Sprint 3: Our biggest concern for this sprint was being able to actually implement the code. We had a lot of features we wanted to implement, and it gets doubled when making them unique for both vendors and customers. However, with proper communication, we were not only able to implement the code, but also test it periodically during development for bugs. Completing user tests also helped us understand that we were mostly missing features to add in the future, rather than functionality bugs. Another major concern for this sprint was being able to test with users. However, we chose to use Expo Go, where we had users scan the QR from our laptops. Instead of installing and running the code locally, they were able to test the app themselves easily.

Sprint 4:

Issue 1: Feedback from the teaching team, peers, testing, and background research all showed that this app must enable real vendors to report locations quickly and efficiently.

Solution: For this sprint, we decided to create personalized reporting and profile pages to streamline reporting. This way, they won't have to input information manually repeatedly. However, this was done without feedback from vendors. Next sprint, we want to get vendor testing so that we can reflect the true needs of busy food truck operators. Securing time with real vendors is challenging because they operate unpredictable schedules and may not have the flexibility to interview or consistently test our product. However, without their feedback, our design risks overfitting to student expectations.

Next steps: Now that we have an iterated app, we will partner with food truck operators who already attend campus events or operate near public spaces to fine-tune features. We plan to approach vendors in person during non-peak hours only to gather contact information and will follow up over email, if possible. We will also incentivize participation by offering statistics from our user testing (as suggested in class feedback) about how simple the app is.

Issue 2: We want to maintain crowdsourced accuracy and map trustworthiness since our platform depends on sighting reports. There is a risk that inaccurate reports could mislead users. If the system shows stale locations or misidentifies closed trucks, new users may lose trust and stop using the app. Since anyone can report sightings, we must filter out noise or malicious activity.

Solution: We introduced confirmation requirements so new sightings are validated by multiple users. A threshold of three confirmations is required before a pin receives a confident status. Pin fading over time naturally removes outdated information. Vendor accounts also carry trusted status through the vendor profile and verification. These mechanisms protect map quality. We will continue testing different confirmation thresholds and visual indicators through A/B testing to see what leads to the highest user satisfaction.

Next steps: Thresholds might need to be dynamic based on how many users use an app in a specific region. For instance, if we have 500 users in NYC vs. 100 users in Atlanta, the thresholds should be higher for NYC than Atlanta. Next sprint, we can conduct testing and create a formula that tracks how many users are registered in a city. Using that, we can dynamically adjust the threshold for the city.

Issue 3: We want to ensure security and authentication due to crowdsourced information. We originally collected emails for this purpose. However, during testing, we realized that we could input an email that wasn't actually real. This would allow users to create spam reports with fake emails, creating trust issues.

Solution: We now have implemented email verification. For users, we have a threshold implemented for reporting security. However, vendor reports are automatically approved, and they only have email verification. This means that as long as they have a valid email, they can sign up to be a vendor.

Next steps: Next sprint, we plan to introduce additional verification for vendors. From our brainstorming, we plan for this to be a manual process. For instance, vendors will have to submit proof that they are a vendor (screenshot of them owning an instagram page, etc.), which will then be sent as a request to our team. Once we approve that, they will get access to create an account. However, we recognize this can be tedious and plan to ask our peers in class about additional ways we can implement extra security on the vendor side.

Sprint 5:

One concern we have is the growing scope. As we continue adding vendor onboarding, admin dashboards, verification flows, and analytics, the amount of work increases quickly. It can be difficult to balance these new features with the need to keep the app stable and usable. To manage this, our team plans to tighten our sprint goals and focus on one core learning outcome at a time instead of spreading our efforts across too many features. This will help us deliver predictable progress and prevent rushed or incomplete implementations.

Another concern is how dependent we are on Firebase and Firestore. While these tools work well for student projects, some administrative actions still feel inconvenient. For example, vendor verification currently requires manual updates in the Firebase Console. This slows down our testing process and creates a barrier to evaluating real workflows. Our plan is to reduce that dependency by building more admin tools directly into our dashboard so that core tasks like approving vendors, viewing proof images, and updating verification status can be handled inside our own interface rather than through Firebase. This should make the system smoother, more secure, and easier to test.

We are also aware that access to domain experts is limited. Food truck owners are busy and not always available for feedback during the week. This makes it harder to validate assumptions about vendor reporting patterns, onboarding steps, and approval processes. To address this, we plan to prepare shorter and more focused testing sessions that can fit into the time that vendors can spare. We also want to recruit a small set of recurring testers so that we receive consistent feedback about improvements across sprints instead of one time conversations.

Another concern relates to scaling our admin backend. We want dashboards, analytics, and approval flows, but all of those require a backend that is reliable and able to handle growing complexity. This is new to our team. To overcome this, we plan to build the backend in small pieces, starting with simple review endpoints before moving toward more advanced analytics. We also plan to write clear documentation for complex analytics so that our team stays aligned as the backend grows.

APA Citations

Pew Research Center. (2021). Mobile fact sheet. https://www.pewresearch.org/internet/fact-sheet/mobile

Allied Market Research. (2023). Food truck market size, share, competitive landscape and trend analysis report. https://www.alliedmarketresearch.com/food-truck-market

2024 statistics of the Food Truck Industry (2024). FoodParks.io. https://www.foodparks.io/blog/2024-statistics-of-the-food-truck-industry

Khuta K. (2025, April 11). *How to develop an app like Waze: process, features, and costs.*Volpis. https://volpis.com/blog/how-to-develop-app-like-waze/

Ettinger J. (2025, Sep 10). *Roaming Hunger App Herds the Mobile Food Truck Masses.* Organic Authority. https://www.organicauthority.com/buzz-news/roaming-hunger-app-herds-the-mobile-food-truck-masses

Food Truck Nation. U.S. Chamber of Commerce Foundation. www.uschamberfoundation.org/solutions/incubator/food-truck-nation.

Garza G. (2022, July 26). *10 must-try food trucks rolling through Atlanta.* Access Atlanta. https://www.accessatlanta.com/atlanta-things-to-do/10-must-try-food-trucks-rolling-through-atlanta/ZIACFHBBCJF3VEP53VH63WDWZ4/

Zuluaga T. (2025). *Food Truck Apps: Best Apps for Food Truck Businesses (2025).* Toast. https://pos.toasttab.com/blog/on-the-line/best-food-truck-apps?srsltid=AfmBOoo1FkzdSAxF_oXRmsag1NoMZnnAwwjNyi1kclHKwUxJxre-qWXg