PRODUCT REQUIREMENTS DOCUMENT v.2

CS Capstone | Fall 2020

Team

	Daniel Kluzner, *Lead*	danielkluzner@ucsb.edu
	Erick Rios, *Scribe*	erickrios@ucsb.edu
	Sai Kathika	saiprem@ucsb.edu
	Huanhua Xu	huanhua_xu@ucsb.edu
	Kelly Yeh	kellyyeh@ucsb.edu

Project Title SmartFarming

Company AgMonitor
Team Name PowWow++

Introduction

PROBLEM STATEMENT

Farming and agriculture play a critical role in the modern world, providing crops for a growing population as well as combating world hunger. According to research, by 2100 the world will need to have increased its calorie intake by eighty percent to feed the population, suggesting farming efficiency will be an important global area of focus. Already with the COVID-19 pandemic, and California droughts in recent years, farming inefficiency is an underlying issue that has affected many people in our surrounding communities.

Additionally, currently in the farm worker hierarchy there is contention between agronomists and ranch managers, due to a lack of data. Agronomists are a third party entrusted with analyzing data in the farms and advising the ranch managers on how to use their resources properly. Because the data is not easily accessible, ranch managers, who are in the field almost everyday, can be skeptical of the agronomist's opinions, especially since the agronomist is not typically in the field everyday. The technology we propose will help agronomists and ranch managers in having an easy to use medium to analyze on-site sensor data and get a grasp on how efficient they are with the farm's resources.

BACKGROUND

AgMonitor is an agriculturally focused software company that creates technology to help farmers keep track of data, manage tasks, communicate with their teams, and schedule upcoming irrigation/fertigation times. AgMonitor provides a plethora of agriculture monitoring software, leveraging their data mining expertise and patented technologies to make products with specific focuses on food, water, and energy, helping users ranging up and down the farming hierarchy. In the context of this project, AgMonitor is focusing on providing the resources for us to expand upon their mobile app.

GOAL/IMPLEMENTATION

Our goal with SmartFarming is to expand upon an existing mobile app using AgMonitor's existing codebase and sensor data to help farms with water efficiency and fertigation issues in hopes of providing an application that can effectively reduce the waste of resources in fields. AgMonitor already has a robust web app as well as a mobile app with limited features. We plan on expanding the current mobile app to include functionality available on the web app, as well as incorporating new features.

One of our main focuses is providing charting features that will display sensor data for different time intervals so that users can be better advised on how their field is using resources and yielding crops. Currently, graphs are available on the web app only, so providing them in a mobile version will greatly benefit farmers. For our chart API we plan on using Victory charts, as this API is relatively simple to use in a React-Native heavy environment, provides endless customization options, and is highly recommended by React-Native developers. The charts themselves will appear in both half screen and full screen modes, features that are implemented seamlessly with React-Native. When a user selects a sensor from the map screen, a simpler snapshot graph with the corresponding sensor data will appear in a half screen mode. From there the user can choose to slide up and view the full graph with data points for whichever time frame they choose, and whichever data field they choose (moisture, temperature, etc.). The time frame and data fields can be selected with buttons, and the full graph will have functionality for zooming and pinching, letting users view data for any interval with a simple gesture.

There will also be a calendar feature in the app that will provide an easy way for farmers to manage upcoming tasks, set alerts, and make sure they are on schedule. Tasks that farmers create on the web app will also show up in the calendar in the mobile app. When the app is booted up, there will be a login screen where users can sign in with their credentials with the help of Firebase. Currently, users can only register via the web application available, but once they have an account they can sign in and use the mobile app. Once they have logged in, users can choose to either view the map or view the calendar. The map screen will allow users to search for their specific farm, or any specific farm they would like to view with sensor data partnered with AgMonitor. When looking at a certain field, users can move/drag around the map and select certain areas of the field to inspect and evaluate, or select certain sensors to view data in charts, as mentioned previously. Users can also search for specific sensors in the map screen search bar. From the calendar screen the user will be able to view/edit their schedule in weekly or monthly intervals.

From a technological standpoint, the app will be coded with React Native for versatility that allows functionality in IOS and Android. The React Native framework is proven to be an excellent tool for creating intuitive user interfaces, and is already used heavily in the current mobile application. The app works in tandem with a Google Maps API to provide the mapping functionality so users can choose which field and which sensor they want to analyze. This Google Maps functionality is also already in use in the current mobile app. In terms of the sensor data and database, Firebase is being used to store all sensor data, and an API for fetching and formatting this data will be developed. The data fetching API will work in two major parts. First, the sensor definitions will be fetched via GraphQl. GraphQl is a useful resource in this context in that it is organized in terms of fields and types, and allows several resources to be attained in a single request, which makes the fetching of sensor data fields (such as name, ID, or units) quick and easy. The second portion of the API will use the previously fetched data fields as arguments and return a tabulated version for use in the charts. An additional hurdle comes with considering the slow loading times associated with large data fetches for the charts, especially when zooming in and changing the time intervals quickly. The way the web app handles this currently is by fetching from the server every time the time interval is changed, but this method is not acceptable for the mobile app. Because of this, we have decided to implement a caching functionality for the data, for faster loading times, with the help of either CouchDB or SQlite, both of which are robust databases with widespread use in offline caching.

CURRENT SOLUTIONS

Agronomists typically use pen and paper to tabulate data when conducting field inspections, or resort to bringing their laptops out onto the field to use the web application. Another way they gather data is by extracting from memory cards inside the on-site sensors. On top of that inconvenience, agronomists are restricted to only looking at the data every two weeks.

As mentioned before, AgMonitor has an existing mobile application that is optimized for irrigators but does not show data that would be useful to ranch managers and agronomists. There are also features that are only available on the web application, such as the charts, which makes it difficult for ranch managers to upload data and edit plans for their crew on the field. There exist a few other competitors that have similar products, but their charts are less intuitive and harder to use.

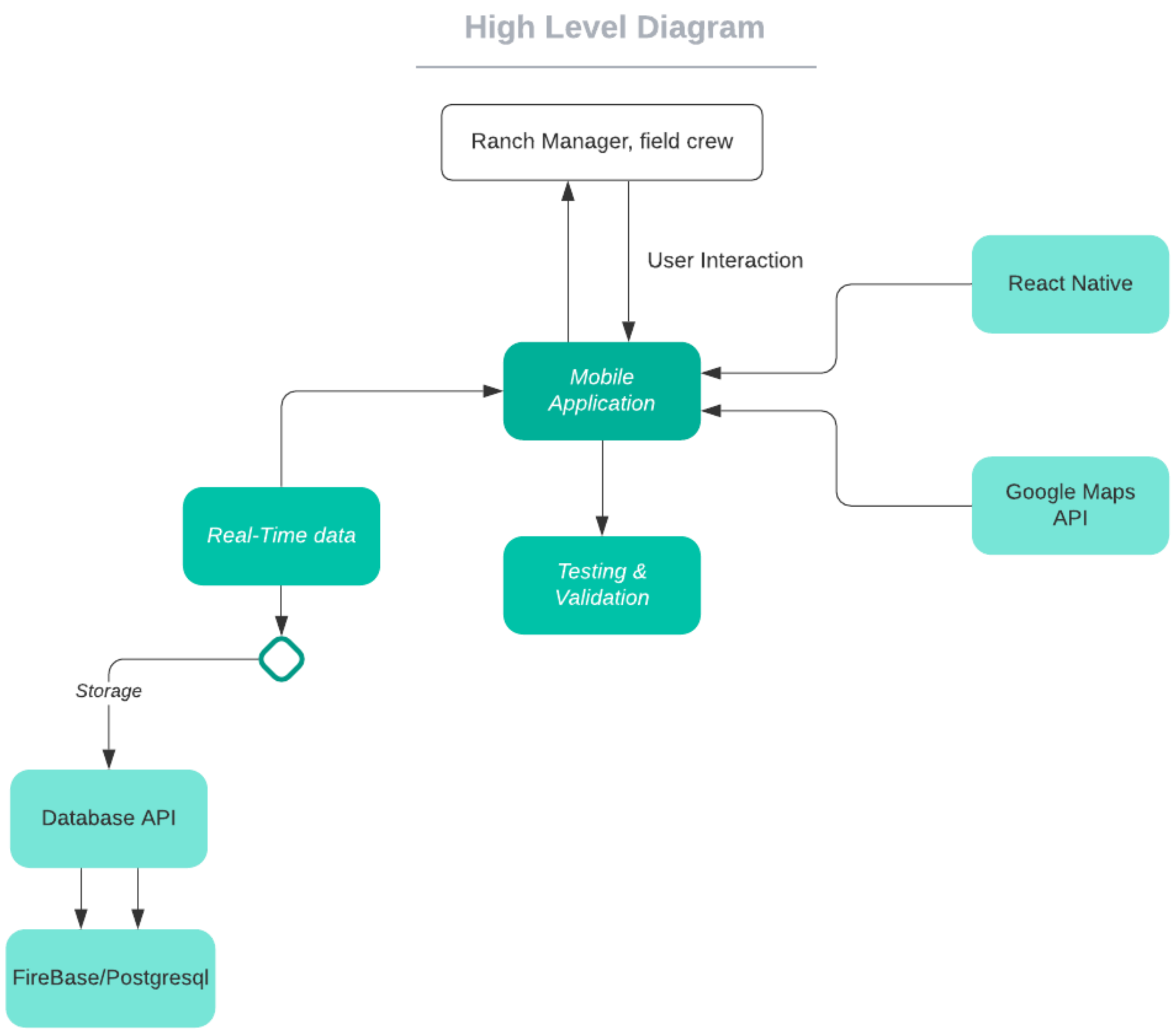
In the context of the current solutions available, a robust mobile application would be greatly beneficial for farm workers, especially ranch managers and agronomists.

Assumptions

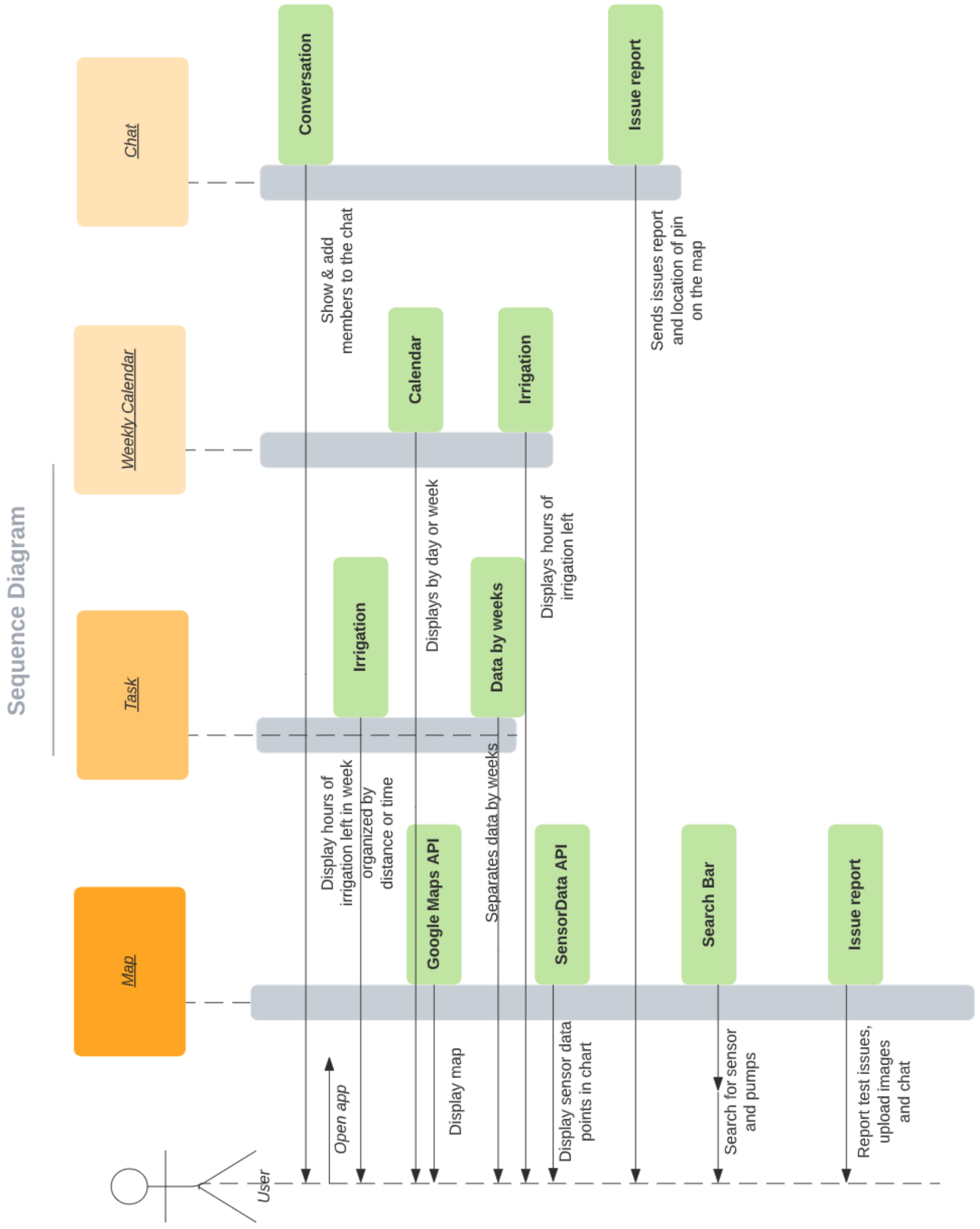
- Our target users will be farm owners and workers in California (only provided data for farms in California)
- Users will need to login and create to access the data
- Data will be available for AgMonitor customers
- Users will be able to quickly comprehend the meaning of the data through charts and other visual tools

System Architecture Overview

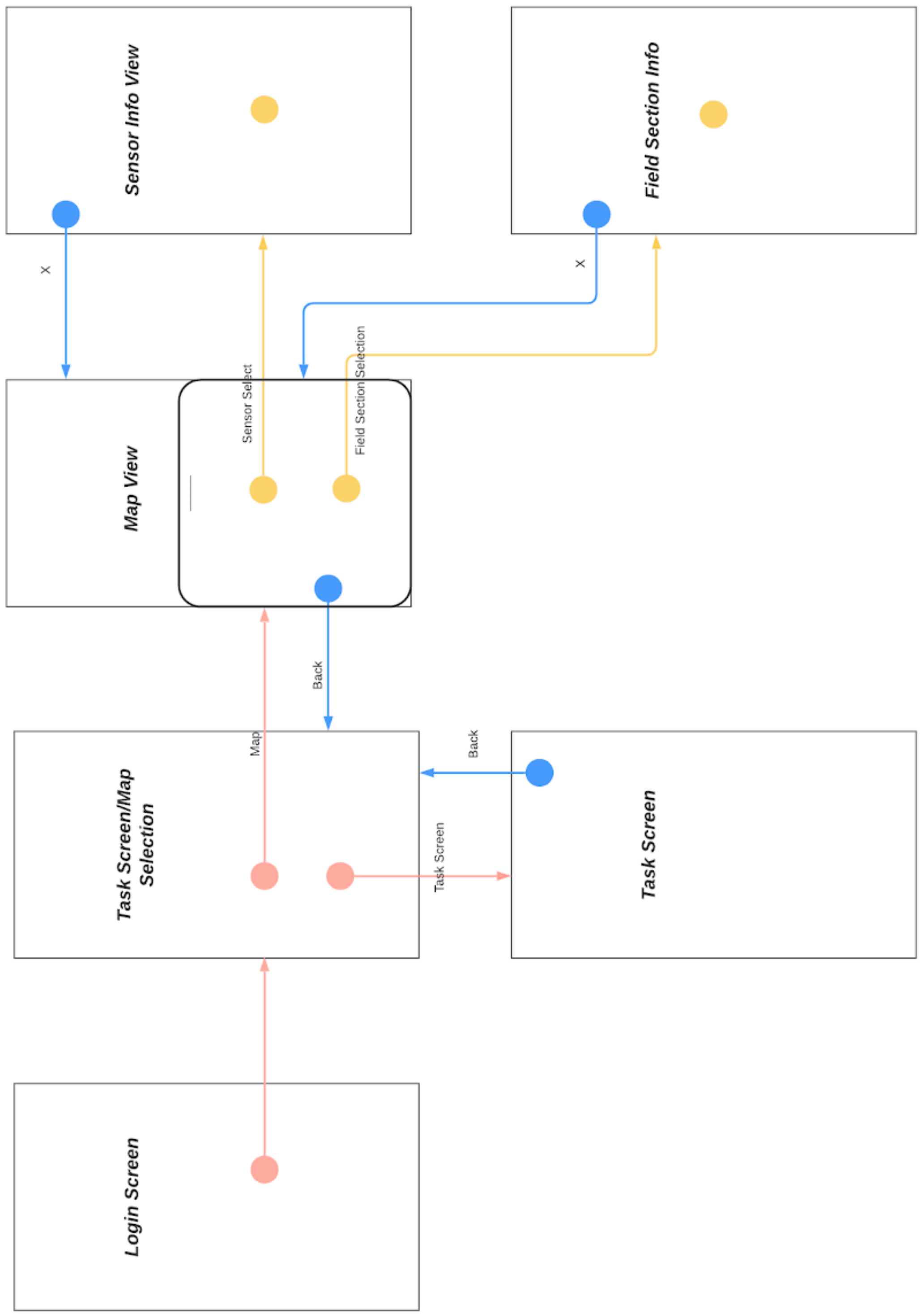
HIGH LEVEL DIAGRAM



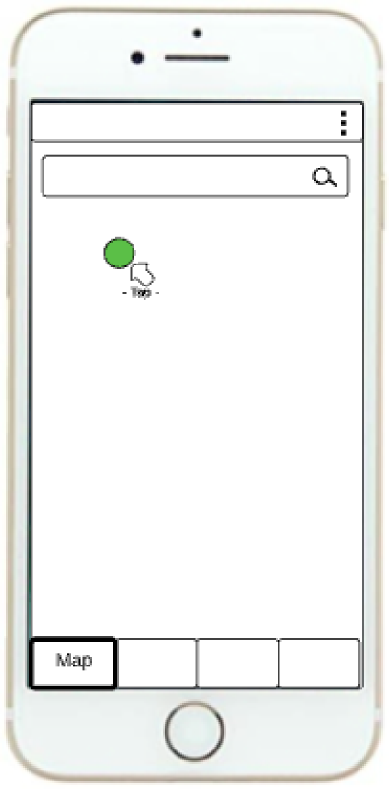
SEQUENCE DIAGRAM

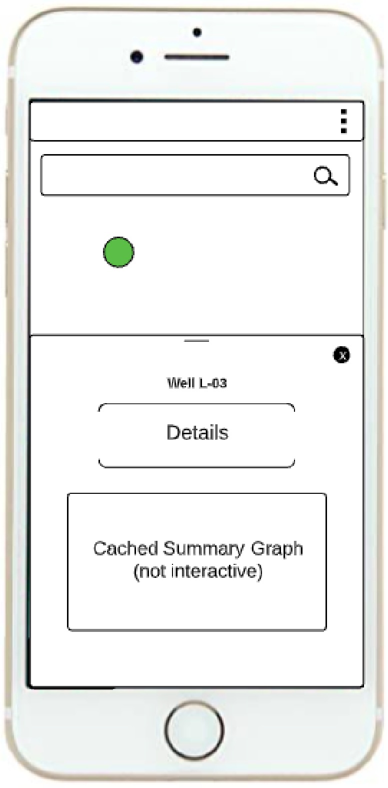

Sequence Diagram
User
Map
Task
Weekly Calendar
Chat
Conversation
Issue report
Calendar
Irrigation
Irrigation
Data by weeks
Google Maps API
SensorData API
Search Bar
Issue report
Open app
Show & add members to the chat
Display hours of irrigation left in week organized by distance or time
Displays by day or week
Display map
Separates data by weeks
Displays hours of irrigation left
Display sensor data points in chart
Sends issues report and location of pin on the map
Search for sensor and pumps
Report test issues, upload images and chat

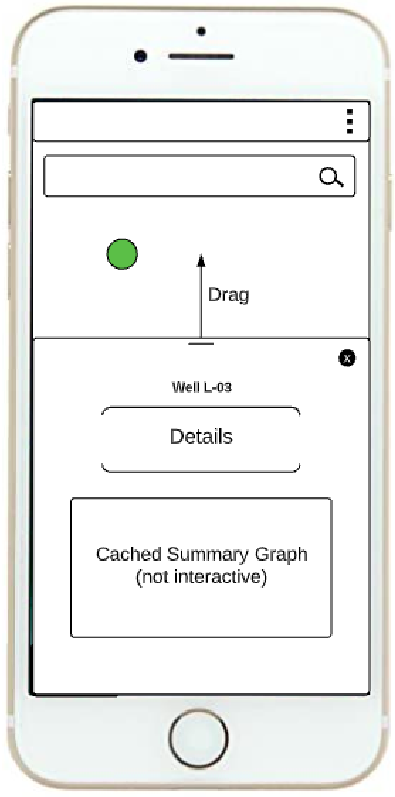
MOBILE APP UI DESIGN

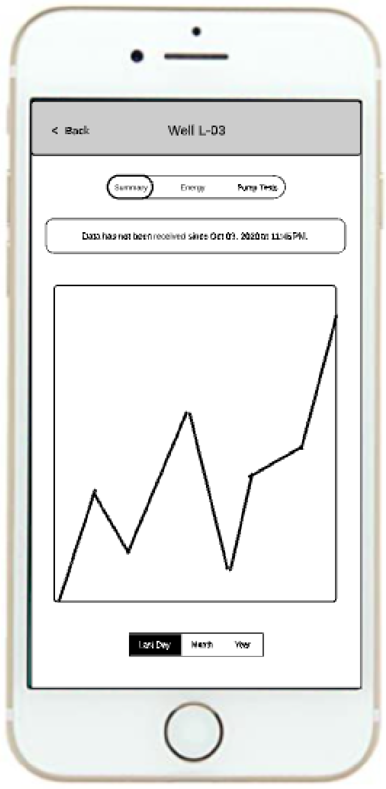


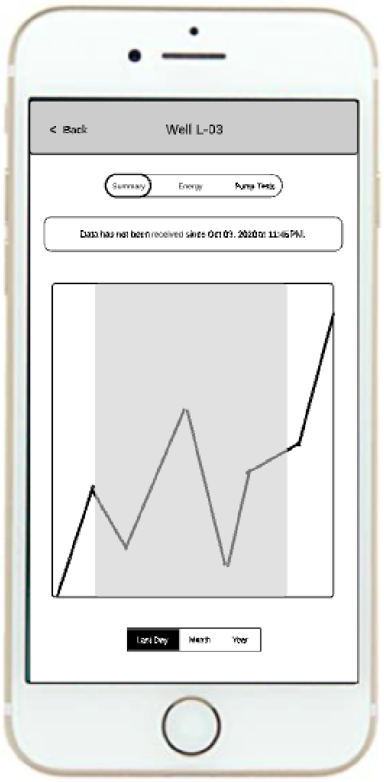
DETAILED UI DESIGN & INTERACTION



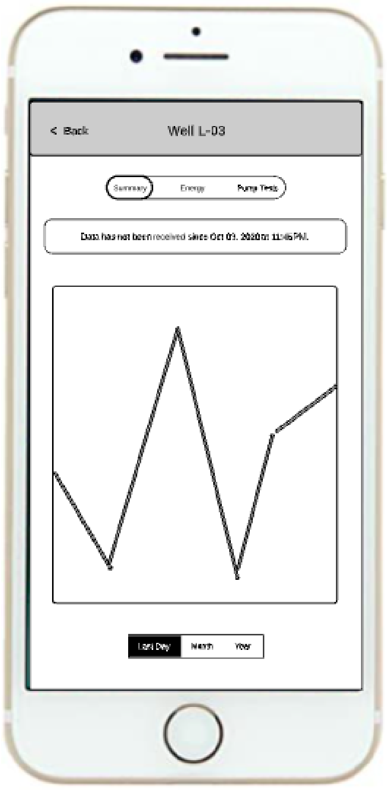




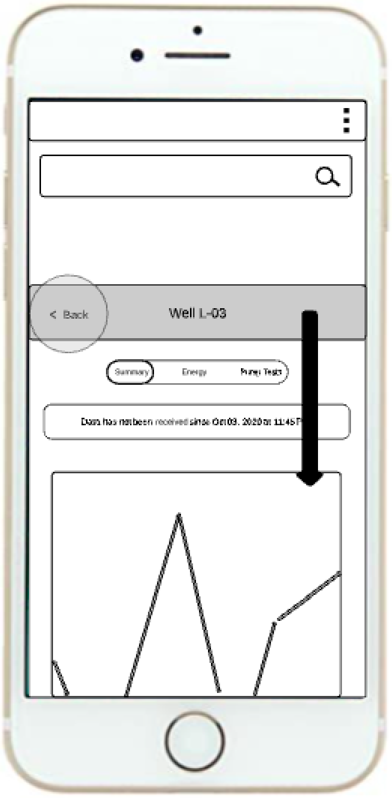




Data has not been received since Oct 03, 2020 at 11:45 PM.



Data has not been received since Oct 03, 2020 at 11:45 PM.



Data has not been received since Oct 03, 2020 at 11:45 P

User Stories

---

1. [*Low priority*] As a ranch manager, I can invite new user and authorize the registration of the new user
    - Acceptance Criteria:
        i. Scenario #1 (New user, but not an employee in the company): Given a new user who want to register an account and the new user is not an employee in the company, when he or she wants to register an new account, then an error message will pop up and say “Need Authorized”
        ii. Scenario #2 (User as ranch manager): Given a user with the authority of ranch manager, when a new employee enter company, the user is able to send an invitation link to the new employee for registration authorized
        iii. Scenario #3 (New user as new employee): Given a new user is an incoming employee in the company, when the user receive an invitation link for registering an account, then the user is able to create a new account

2. As a registered user, I can check the information I saved (specific map area, specific sensors on the field, etc.)
    - Acceptance Criteria:
        i. Scenario #1 (User with registered account): Given the user already registered an account, when the user logs in to the application with correct username and password, then the user can check their saved information
        ii. Scenario #2 (User with registered account): Given the user already registered an account, when the user denotes a specific area or sensor as a ‘favorite’ item, if something is updated inside the favourite items, the the user will receive notification
        iii. Scenario #3 (User without account): Given the user hasn’t created an account yet, when the user tries to login, the user will be prompted to sign up first

3. As a user, I can zoom in on the maps and check specific sensors on the mobile phone so that I can get the real-time data about local situation
    - Acceptance Criteria: After logging in to the application, the user can select a specific field to view as a map. The map will allow users to zoom in or zoom out, and drag to move the view. Also, when the user taps on a specific senor on the map, the user is able to check the data for that sensor

4. As a user, I can choose different time intervals (last day, last week, last month, last year) and view the data corresponding to that interval
   - Acceptance Criteria:
     i. Scenario #1 (viewing data in day interval): Given a specific sensor, the user can click the tab named ‘Day’, then the data on the chart will show in days interval
     ii. Scenario #2 (viewing data in week interval): Given a specific sensor, the user can click the tab named ‘Week’, then the data on the chart will show in weeks interval
     iii. Scenario #3 (viewing data in month interval): Given a specific sensor, the user can click the tab named ‘Month’, then the data on the chart will show in months interval
     iv. Scenario #4 (viewing data in year interval): Given a specific sensor, the user can click the tab named ‘Year’, then the data on the chart will show in years interval
     v. And so on.

5. As a user, when viewing a chart I can zoom in and view the data for smaller intervals of time, depending how much I zoomed in
   - Acceptance Criteria:
     i. Scenario #1(Zoom in): Given the user viewing a chart, when the user zoom in the chart by pinching the screen with two fingers, the intervals of time on the chart will be smaller so that the user can view more specific information during certain period on the chart
     ii. Scenario #2(Zoom out): Given the user viewing a chart, when the user zoom out the chart by pinching the screen with two fingers, the intervals of time on the chart will be larger so that the user can view the overall change during a long period on the chart

6. As a user, I can choose what kind of data to display on the graph (soil moisture, temperature, humidity, pressure, etc.) so that I can clearly see the trend of development
   - Acceptance Criteria:There are tabs that indicate different variables, and the user can press any of them so that the data of the chosen variable will be shown on the graph

7. As a user, I can check the hours since the last irrigation and show irrigation status
   - Acceptance Criteria: The user can select a specific sensor. By clicking on the sensor, it will show the irrigation schedule and the water use

8. As a ranch manager, I can view and edit the tasks on the calendar., and any update of the calendar task will notify the group member through the mobile application
    - Acceptance Criteria: if the user is an admin, then the user can update the task on calendar by adding new events, then it will notify the group about the update

9. As a user, I will be notified of updates if the ranch manager has altered any tasks on the calendar
    - Acceptance Criteria: A user with admin status will be able to alert other members of their group when they update the calendar

10. As a user, I can search for a specific area or sensor and display their data
    - Acceptance Criteria: By entering specific information such as serial number on the search bar, the user can directly access the data of a specific area or sensor. If the information and serial number don't exist, it will return an error message and prompt the user to enter correct information

11. As a user, I can observe if the water is used inefficiently and adjust the water use
    - Acceptance Criteria: After the user clicks on a specific sensor they would be able change the irrigation patterns for more optimal water efficiency

12. As a user, I can see the soil moisture level and history of recent irrigation in the field containing the sensor on a single graph
    - Acceptance Criteria:
        i. Scenario #1 (By clicking or searching a specific sensor): Given a specific sensor, the user can click the sensor on the map or search the sensor through search bar, the user is able to view the soil moisture level and history of recent irrigation on a single graph
        ii. Scenario #2 (By entry of irrigation events): Given an irrigation event, the user can search for a specific irrigation event, then the user can view the soil moisture level and history of recent irrigation events for this relevant sensor.

13. As a ranch manager, I can create new calendar items for manually irrigated fields, or edit existing events, and save these changes and have them go to the field crew
    - Acceptance Criteria:
        i. Scenario #1 (create new calendar event): Given a user with admin privileges, when the user clicks the "new event" button on the UI interface, the user is able to create new calendar items for manually irrigated fields and save the changes on calendar which will automatically notify the field crew.

ii. Scenario #2(edit existing items): Given a user with admin privileges, when the user clicks the “Edit” button on the UI interface, the user is able to edit the existing events and save the changes, and have them go to the field crew

14. As a ranch manager, I can see the data from my weather stations as a graph
    - Acceptance Criteria:
        i. There is an option to show two different metrics in two separate graphs
        ii. The different metrics vary by season, examples include: temperature, humidity, rainfall, wind speed, wind direction

15. As a ranch manager, I can see my soil moisture with weather data (especially rainfall or temperature) to monitor soil moisture in certain months in full screen mode
    - Acceptance Criteria:
        i. User can also see the history of recent irrigations in the field containing the sensor by tapping on the corresponding field

16. As a user, I can slide up the half screen chart so that I can view the data in full screen
    - Acceptance Criteria:
        i. Scenario #1 (viewing the summary of data): Given a specific sensor on the map, while the user clicks on the sensor, a half screen chart will shows up to display the summary of the data
        ii. Scenario #2 (viewing the detail of data): Given a specific sensor on the map, while the user clicks on the sensor and slide up the half screen chart, a full screen chart will display a graph showing the data in different metrics and time intervals

17. As a user, I can send messages to other users working on the same farm
    - Acceptance Criteria:
        i. Scenario #1 (users in the same group): Given a user joining a work group, when the user tries to send a message to notify fellow field workers or agronomists, the other members in the group with receive the notification

18. As a user, I can “create an issue” from the sensor data overlay (half- or full-screen)
    - Acceptance Criteria: When a user wants to report an issue occurring with either a sensor, pump, or well, they can click the corresponding button in the sensor overlay and it will link to a conversation in the chat where they can forward the issue to a supervisor or a fellow worker

19. As a user, I can take photos and attach images to the issue report

    - Acceptance Criteria: The workers reporting the issue can attach an image that shows what the problem is, and this photo will be relayed in the chat message along with the issue report

20. As a user who receives a message reporting an issue, I can locate it on the map
    - Acceptance Criteria:
        i. The message received will have linked to it the issue, and a button "Show on map" will take the user directly to the corresponding location on the map
        ii. If an issue is not linked to a specific issue, "Show on map" will not show up

Project Specific

FRONTEND

- Using React Native to develop on Android and iOS platforms simultaneously
- Implement login page and registration page where users can enter credentials to enter into main interface of application
- The charts will be displayed on the frontend using a React Native API
- Chart API that we plan on utilizing: Victory or React-Native-Charts
- The app will enter the frontend through App.js where it will be routed to different screens— Login/Register based on users actions

BACKEND

- We do not have access to the backend yet but we presume it be Django to interact with the database
- The backend will interact with the frontend, pull data from the database, and perform all the machine learning analysis and calculations necessary

DATABASE

- Will be using Firebase along with Firestore as our database
- Currently stores various sensor data
- Research an API that can automatically format and consolidate all of the data to be stored in database

LOGIN AUTHENTICATION

- Using existing AgMonitor mechanisms that use Firebase Custom JWT created in the server
- Created persistent login feature where users will not have to sign in again after they exited the app
- Postgres backend stores username, email, and hashed password

Appendix A: Technologies Used

DATA STORAGE AND WEB APP

- PostgreSQL
- Firebase
- Python, NumPy, Pandas, Scikit-Learn
- Django, Flask
- Javascript

MOBILE APPLICATION

- React Native
- Android, iOS

OTHER TOOLS

- API's: Google Maps
- Docker
- Windows WSL
- LucidChart
- Firestore
- Android SDK

When the app is booted up, there will be a login screen where users can sign in with their credentials with the help of Firebase. For chart API's we plan on using either Victory or React-Native-Charts, as both these API's are relatively simple to use in a React-Native heavy environment and provide endless customization options.

From a technological standpoint, the app will be coded with React Native for versatility that allows functionality in IOS and Android. The React Native framework is proven to be an excellent tool for creating intuitive user interfaces, and is already used heavily in the current mobile application. The app will work in tandem with a Google Maps API to provide the mapping functionality so users can choose which field and which sensor they want to analyze. This Google Maps functionality is also already in use in the current mobile app. For the sensor data, API calls will fetch data from our firebase database, using either Flask or Django. Flask and Django are both Python frameworks that help with handling large amounts of data transfers in a quick and efficient manner. Firebase will store all our sensor data.