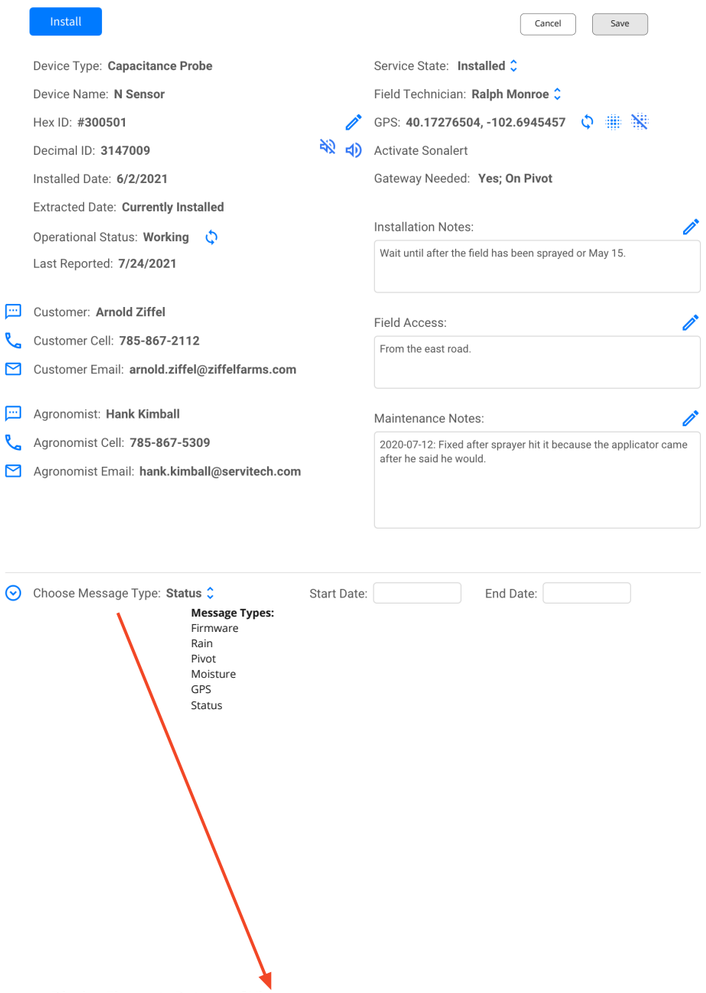
device-detail-view

Display Elements Left side
Create code that defines elements which display and/or allow edit data values for an object in "allData" identified using the integer selected in "objectSelector" as follows:

- **"Install" button and event handler** to:
  - if Device Type = "Gateway" pop modal "Install Gateway" view
  - Else pop modal "Install Device" view
- Label to display **Device type**
- Label to display **device name**
- Label to display **hexID**
- Label to display **decimal ID**
-
- Label to display **installed date**
- Label to display **extracted date**
- Label to display **operational status**
  - **refresh icon** to send request to server to update status
  - refer to current bridge dashboard check device button
- Label to display **last reported**
- Label to display **customer name**
  - **chat icon** to send sms message (modal popup)
- Label to display **customer cell phone number**
  - **phone icon** (may not be needed) link for click to call
- Label to display **customer email address**
  - **email icon** to send an email
- Label to display **agronomist name**
  - **chat icon** to send sms message (modal popup)
- Label to display **agronomist phone number**
  - **phone icon** (may not be needed) link for click to call
- Label to display **agronomist email**
  - **email icon** to send an email



<Task Assignment: device-detail-view>
Project elements provided:

- Basic React Project
  - google map (see illustration)
- geoJSON files
  - deviceLocations.json
- Simple CSS
- set of marker icons (svgs)
- Illustration "*map-view*"
- Instructions
- Description of Behavior
- Coding Standards

Instructions (ref. Illustration *install-device-view*):

- Reference the provided project and illustration
- Insure the project works in your local (dev) environment (npm start)
- open the app.js and note the section marked "where the components are"...
- In the file **"device-detail-view.js**" find a section marked "your code goes here..."
- **The provided project includes:**
  - code that maps a feature collection object **allDeviceLocations** from **deviceLocations.json** as a state property
  - a **Google Map component** with a default data layer pre-loaded with objects from the **allDeviceLocations** feature collection
  - code that initializes a **selectedObject** state property to manage device filter criteria
  - code that initializes a **devicesFilter** state property to manage device filter criteria
- Using the illustrations and Behavior detail as references, provide code to define the elements and controls as specified in the instructions provided here.

Display Elements Bottom of View
Create code to define elements as follows:

- **deviceMessageTable expandable section** with "expand/collapse" icon
  - **(expand) icon onClick event handler** that calls fetchDeviceMessages
  - **(collapse) icon onClick event handler** to collapse the section
- selector control to select device **message type** (default = "Status"
- Input Element (date selector) for **Start Date** (default = today - 1 week)
  - onChange event handler that calls fetchDeviceMessages
- Input element (date selector) for **End Date** (default = today)
  - onChange event handler that calls fetchDeviceMessages
- **fetchDeviceMessages function** to accept values (Device ID, Message Type, Start & Stop dates) and:
  - (Using the passed-in values) call appropriate API to fetch message data from devices data source
  - map the returned message data to create a table and show in the deviceMessageTable section
  - set the deviceMessageTable to "expanded" to show table

Display Elements Right side
Create code to define elements as follows:

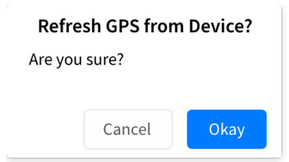
- label and textbox to display **Operational Status**
- selector control to display/select **service state**
  - create **onChange event handler** to update this object in the **allDeviceLocations** feature collection
- selector control to display/select **field tech**
- Label & textbox to display **device GPS**
- **Edit GPS icon & onClick event handler** to:
  - **set map to "edit" mode**
    - this allows the selected map marker to be repositioned
  - create code such that moving the marker on the map:
    - updates the "edit GPS view" textbox with the new lat/lon for the selected object
  - **present model "Set Device GPS" view** with save & cancel buttons
    - create onClick event handler for this view's save button to:
      - update the selected object lat/lon property value with the view's gps textbox
      - set map to "navigation" mode
      - close the "edit GPS view"
    - create onClick event handler for the view's cancel button to:
      - set map to "navigation" mode
      - close the "edit GPS view"
- **refresh icon & onClick event handler** to:
  - read the device GPS from server
  - activate the "**Update GPS from Device Readings**" verification popup (see illustration) with cancel and OK buttons
    - OK button event handler to
      - update allDeviceLocations feature collection object with new readings
      - (this should refresh the deviceGPS text box
      - close popup window
    - Cancel button event handler to close popup window
- **activateSonalert icon** with Label & (empty) textbox (font color = red)
  - **onClick event handler** for icon to:
    - send activate sonalert command to server
    - set textbox.text to "ACTIVATED"
- **clearSonalert icon and onClick eventHandler** to set activateSonalert textbox.text to ""
- **gatewayNeeded** Label & textbox to display gateway needed property value
- **"signal" icon & onClick event handler** to toggle gateway radio range on map around device
  - if exists objectID = this deviceID+"-range" in allDeviceLocations data layer:
    - remove object from data layer
  - else
    - create a circle feature object with center at this device location & 1/2 mile radius and assign an object id = device id +"-range"
    - add feature object to allDeviceLocations map data layer
- Input Element (paragraph box) for **installation notes**
  - **Edit icon** to put data control in edit mode with save/cancel buttons
- Input element (paragraph box) for **field access notes**
  - **Edit icon** to put data control in edit mode with save/cancel buttons
- Input element (paragraph box) for **maintenance notes**
  - **Edit icon** to put data control in edit mode with save/cancel buttons

Date/Time	snr	rssi	intern_temp_c10	frmver	charge_v1000	sensor_status	vcc_v1000
2022-01-21T14:27:53.000Z	3	-118	-9.9	3.1.0.17	655.3	0	**3.35**
2022-01-21T14:25:07.000Z	-4.75	-121	-10.8	3.1.0.17	655.3	0	**3.347**
2022-01-20T16:30:54.000Z	0.25	-118	-6	3.1.0.17	655.3	0	**3.35**
2022-01-19T17:36:32.000Z	3.25	-116	3.9	3.1.0.17	655.3	0	**3.353**
2022-01-18T17:36:32.000Z	5.25	-114	18.3	3.1.0.17	655.3	0	**3.135**
2022-01-17T17:36:32.000Z	5.75	-112	15.1	3.1.0.17	655.3	0	**3.122**
2022-01-16T17:36:32.000Z	4.5	-116	16.5	3.1.0.17	655.3	0	**3.102**
2022-01-15T17:36:32.000Z	4	-117	3.2	3.1.0.17	655.3	0	**3.115**
2022-01-14T17:36:32.000Z	3.5	-115	17.4	3.1.0.17	655.3	0	**3.253**

Service State values

- Not Ready
- Ready to Install
- Installed
- Needs Repair
- Replace Batteries
- Replace SID
- Replace Gateway
- Replace Device
- Ready to Extract
- URGENT Extract
- Extracted
- Service Ready to Perform
- Service in Progress
- Service Complete

should this be set up as an 'admin settings' list that can be edited by a system administrator?



Behavior:

- Referencing url presents this view in a browser
- Upon accessing the URL, view elements appear showing the text from the index 0 data object from the allData array
- Changing the objectSelector control changes the data displayed in the other element controls to reflect data from the object index represented by the integer in the selector control.
- Clicking Scan Code button invokes camera in barcode scan mode, updates data in hexID text box
- Clicking Use Phone GPS reads user-device gps and updates Lat and Lon
- Clicking Read Device GPS button updates the Lat & Lon text controls with data from the Realm Device readings
- Save closes the view and pushes updated data back to the persistence model