1. Build communication back and forth between the google sheet and the web page. A model for that is “WorkingFormToSheet” in my google sheet communicating with [file:///C:/Users/leon/Documents/BB/CDL19/Web/Index.html](file:///C:\Users\leon\Documents\BB\CDL19\Web\Index.html)
2. The google script attached to the google sheet has to respond with data. I need to delve.
3. I need to bring in the player data. See <http://lfeder.com/nrl/load-demo.html>
4. Also, for this, we’ll want to use PapaParse: <https://github.com/mholt/PapaParse> and also <https://www.papaparse.com/docs>
5. We’re going to want to use that list for an autocomplete on the player name. Should be fun!
6. I use the team and slot to determine column and row. This will be in the Google Script
7. I need to see it the slot is filled. If it is, I have to save off the current content of the slot and ask where it moves. I keep doing this until there’s nothing to save off.
8. I need to enable auto-save on the sheet.
9. I want to pass Google-Sheet info back through the ajax call. This will be reused but the first thing is the saved off info about the contents of the current slot.

Pretty good progress over the weekend. Let’s task things up now….

1. Now that I can write and return data, let’s update the sheet by putting a player in a slot. This is mostly about the **Google Sheet Script**.
   1. We’ll overlay first. Whatever we have gets stored there.
   2. We’ll write code to store the value of the cell in an array. That will be passed back to the web page and it’s scripts in the return value. If it’s empty, we’ll know there’s nothing more to do. This will create this type of dialogue:
      * Send: Draft "Freddy Freeman|04-Federal Express|32|L3|1B1|"
      * Reply: Occupied “Cody Bellinger|04-Federal Express|7|L1|1B1|"
      * Send: Draft “Cody Bellinger|04-Federal Express|7|L1|OF1|"
      * Reply: Occupied “Aaron Hicks|04-Federal Express|2|L2|OF1|"
      * Send: Draft “Aaron Hicks|04-Federal Express|2|L2|U|"
      * Reply: Success “….”
   3. The next tasks in the Google Sheet Script are should-have enhancements:
      * Return the entire draft data back to the web page in a table.
      * Record the status of the draft in locked cells so that anyone can use the web application.
2. The web page index.html needs beautification. It needs a layout. It needs structure. It needs big bold displays. We will be using Bootstrap tabs.
3. The google-sheet.js will utilize jquery to handle element manipulation.
   1. Painting into the interface I’ll use Bootstrap 4 tabs and display:
      * Specific team rosters
      * League-wide Budget chart
      * Team need Tracker
      * Time out Tracker
      * Nomination Tracker
   2. Consume the player list
   3. Provide autocompletion of player names