**Project Scope**

**Overall Goal**

Recreate shop scheduling application with a better-looking interface, persisting all current functionality. TBD for certain where this will be hosted – but in any case the server must use TLS version 1.2. Deadline to have 1.2-compatible version is 5/21\*.

**Brief Description of current functionality (1.0)**

* 6 total “views”, which view is shown is determined by URL
* 1 of the 6 is just read only look at the overall schedule
* The other 5 are station-specific and need to have the ability to ‘mark things done’ on the schedule, which is a Boolean in Netsuite
* Each view is sourced from a saved search in Netsuite, containing all information about those work orders
* Each work order can be drilled into to show information about the components (read only)
* All views refresh automatically every 3 minutes and display the last updated time
* Currently this is done with PHP and SOAP-based web services request

**2.0 Version**

- Will utilize Restlets, not SOAP, in Netsuite to provide endpoints and act as a REST API to POST or GET data.

- Token-based authentication (OAuth 1.0) will be used to make API calls

- Server will run off of Node.js – beyond that the technologies utilized are open ended. Bootstrap is certainly fine. React, if warranted.

- Develop in shared GitHub repository

**Additional Features**

* Just need to handle any errors on the API calls and make sure that there is no downtime in the case of error, including concurrency issues. User-friendly error messages would be helpful if we can predict what the potential errors are.

**Leo Responsibilities**

* Create URL endpoints for API connection, format data returned as necessary
* Test to make sure functionality meets requirements
* Provide token keys and secrets which will be used for OAuth
* Clarify the logic or current functionality as necessary

**Chris Lynam Responsibilities**

* Create a new design for how to display the data. Should look professional but the look of it is not critical to project success.
* Utilize node.js to create authenticated requests (this has been done successfully in Postman, just need to do it in Node)
* Allow for adding additional columns of data or manipulation of how data displays in the future.
* Assist in determining how and where to host the application. Heroku might be a simple option.

\*5/21 should be the planned completion date but it may be possible to pick our own deadline if we can get the existing system up to speed on TLS 1.2. Leo will research.

Chris Lynam fee for project: $1,500, due on completion. Chris will provide an Invoice.

Leo Galey IV \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_

Chris Lynam \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_