NAU NORTHERN ARIZONA UNIVERSITY

School of Informatics, Computing, and Cyber Systems

Control System for a Robotic Spectrograph at Lowell Observatory

Team: Henry Fye, Nhat Linh Nguyen, Jacob Penney, Jakob Pirkl, Kadan Seward

Client: Dr. Joe Llama & Dr. Gerard van Belle, Lowell Observatory, Flagstaff, AZ

Team Mentor:
Rudhira Talla

Problem Overview

Introduction to Spectrographs

Professional astronomers have the important but difficult task of gathering information about astral bodies, objects they can scarcely conveniently observe. Among the many tools they use for this task, **spectrographs** allow them to learn about the qualities of these bodies, such as chemical composition, density, temperature, and speed of movement, among others. This information can yield important discoveries, such as **habitable planets**, **black holes**, or **the sources and machinations of strange phenomenon**.

Problems and Our Role

Our clients at Lowell Observatory use some of the finest commercial spectrographs in the world to conduct research just like this every night, but these tools can be unwieldy to use because of their complexity. Software solutions exist which abstract away these complexities, but they are **dated**, oftentimes **break**, and cannot be readily improved upon by those that use them because **their source code is proprietary.**

Feasibility

Website

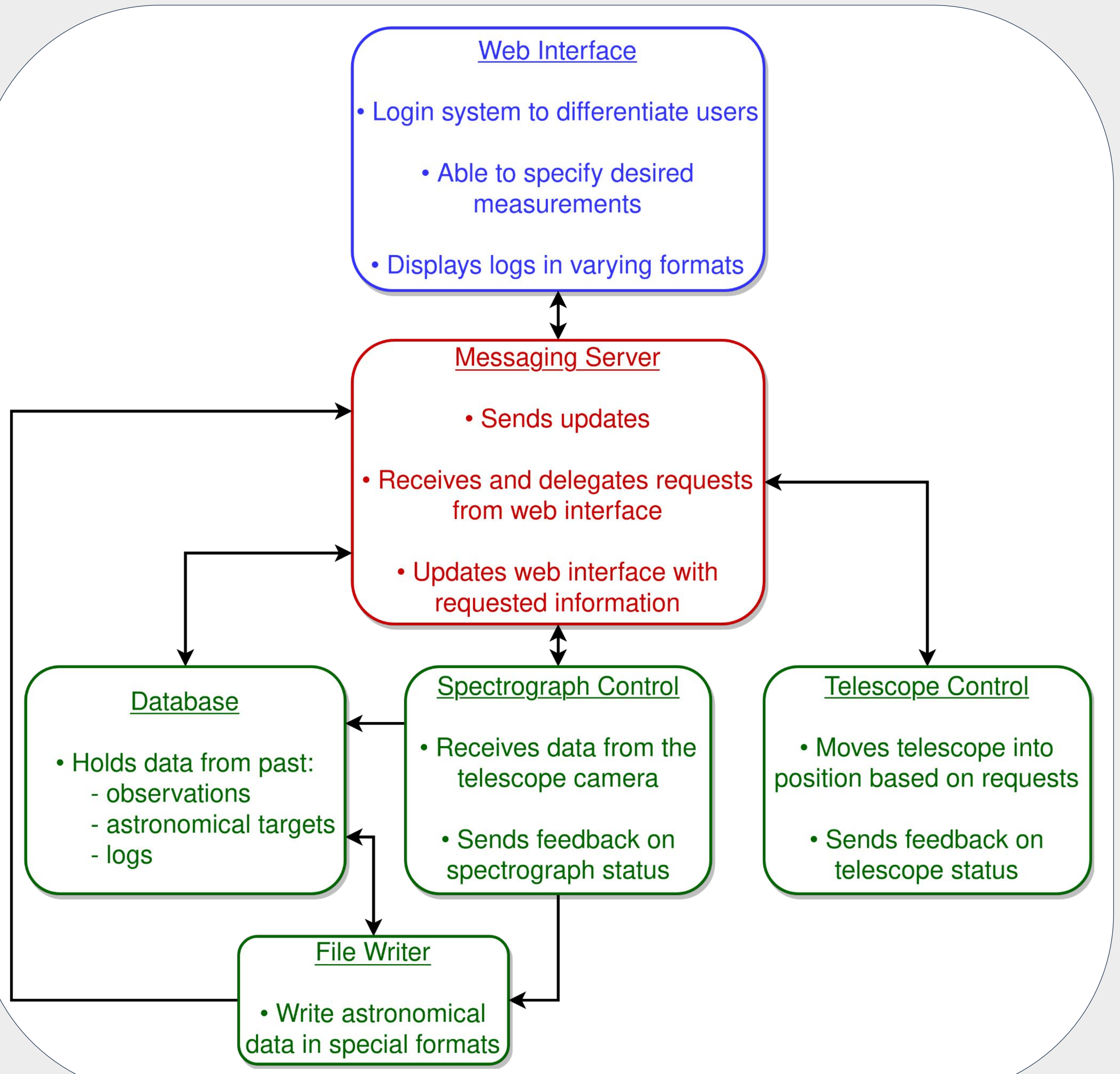
For the frontend, **ReactJS** was chosen. This is because the sponsor is familiar with this frontend as well as the longtime stability of the product. Angular was also considered, but Angular2+ is much newer and has many more features than is needed, leading to a potentially bloated system.

Backend API

Like with the frontend, the backend this project requires is the most simple Python tool able to complete the task. Out of Flask and Django, the most popular of the Python backends, **Flask** was found to be the most simple. Flask was also chosen because several members of the team already have experience in Flask.

Database

When searching for a database, there were two main options: MySQL and PostgreSQL. Though MySQL is often used in enterprise, by companies like NASA, we chose **PostgreSQL** due to the paid licensing of MySQL conflicting with the open source nature of the project.



Web Interface
- Login system to differentiate users
- Able to specify desired measurements
- Displays logs in varying formats

Messaging Server
- Sends updates
- Receives and delegates requests from web interface
- Updates web interface with requested information

Database
- Holds data from past:
  - observations
  - astronomical targets
  - logs

Spectrograph Control
- Receives data from the telescope camera
- Sends feedback on spectrograph status

Telescope Control
- Moves telescope into position based on requests
- Sends feedback on telescope status

File Writer
- Write astronomical data in special formats

Goals and Extensions

Main Goals

The minimum viable product is a basic **web application** composed of a **website** with a login system, a verbose **backend API**, and a **database** capable of storing spectrograph data. This proposed web app will solve our client's workflow issues by allowing them to calibrate and command the spectrograph remotely and subsequently view and operate on gathered data, all via the same interface.

Extensions

Useful potential extensions to the minimum viable product include support for **flexible visualization**, such as zooming in on regions of interest from previous observations via an observation interface; **dynamic database searching**, such as being able to see observations from previous nights using a built-in search interface; and **timed release of gathered data to the greater scientific community**.

Envisioned Solution

Website

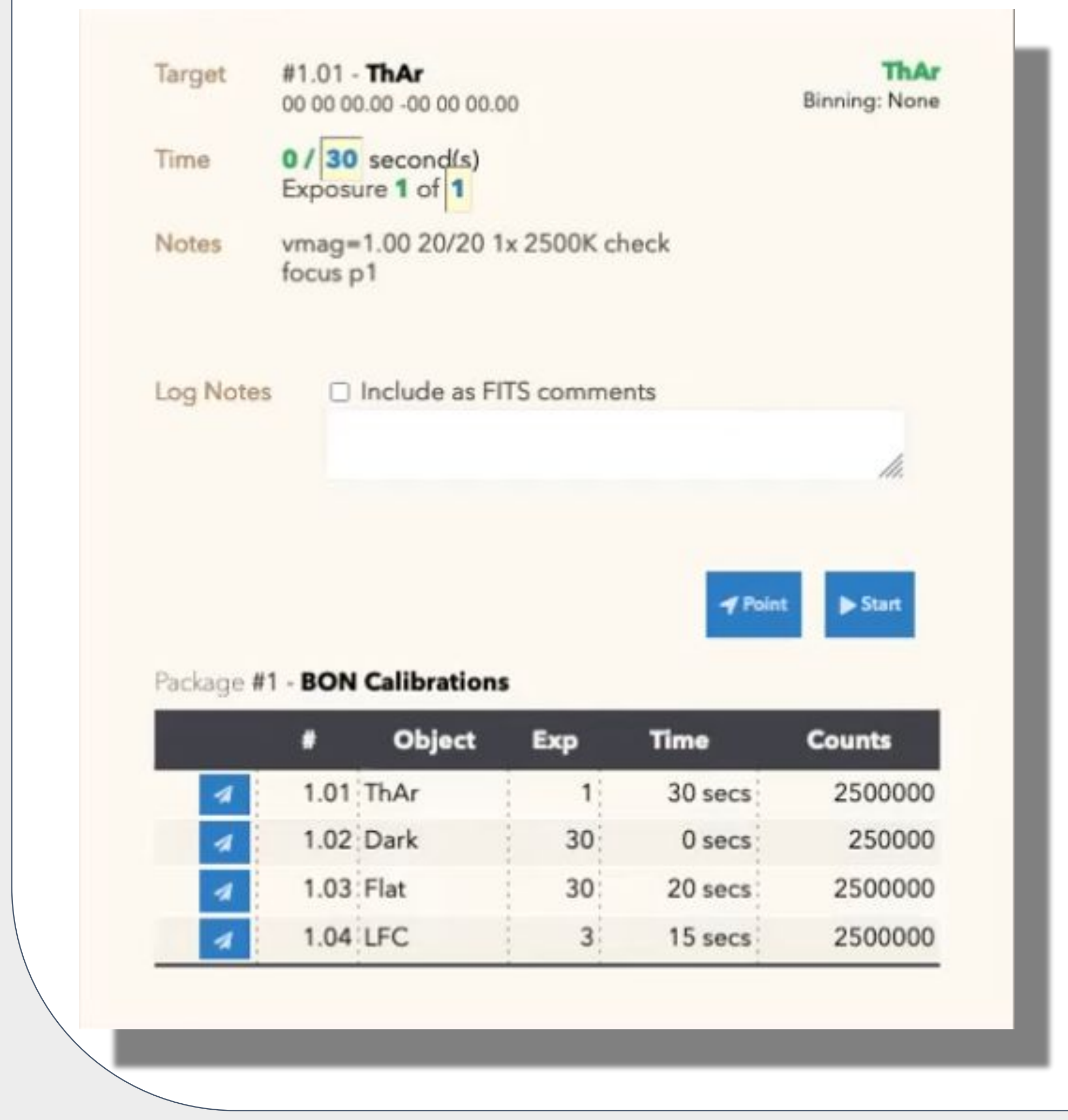
A friendly, easy-to-use **interface** provides live status updates from various **subsystems**, a table of prior observations, and information about ongoing observations.

Backend API

A **server** responsible for sending parameters which can be interpreted into commands that the spectrograph can use. The data the spectrograph collects can be returned here for dispatch to the frontend.

Database

A **SQL database** which stores logs of the various observations that have been taken and data about targets that have been observed.



Target #1.01 - ThAr
00 00 00.00 -00 00 00.00

ThAr
Binning: None

Time 0 / 30 second(s)
Exposure 1 of 1

Notes vmag=1.00 20/20 1x 2500K check
focus p1

Log Notes ☐ Include as FITS comments

Package #1 - BON Calibrations

	#	Object	Exp	Time	Counts
	1.01	ThAr	1	30 secs	2500000
	1.02	Dark	30	0 secs	250000
	1.03	Flat	30	20 secs	2500000
	1.04	LFC	3	15 secs	2500000

Figure: Example user interface for selecting a celestial object and the desired duration of the spectrograph's exposure to it.

Technologies Planned

React JS

Flask

PostgreSQL