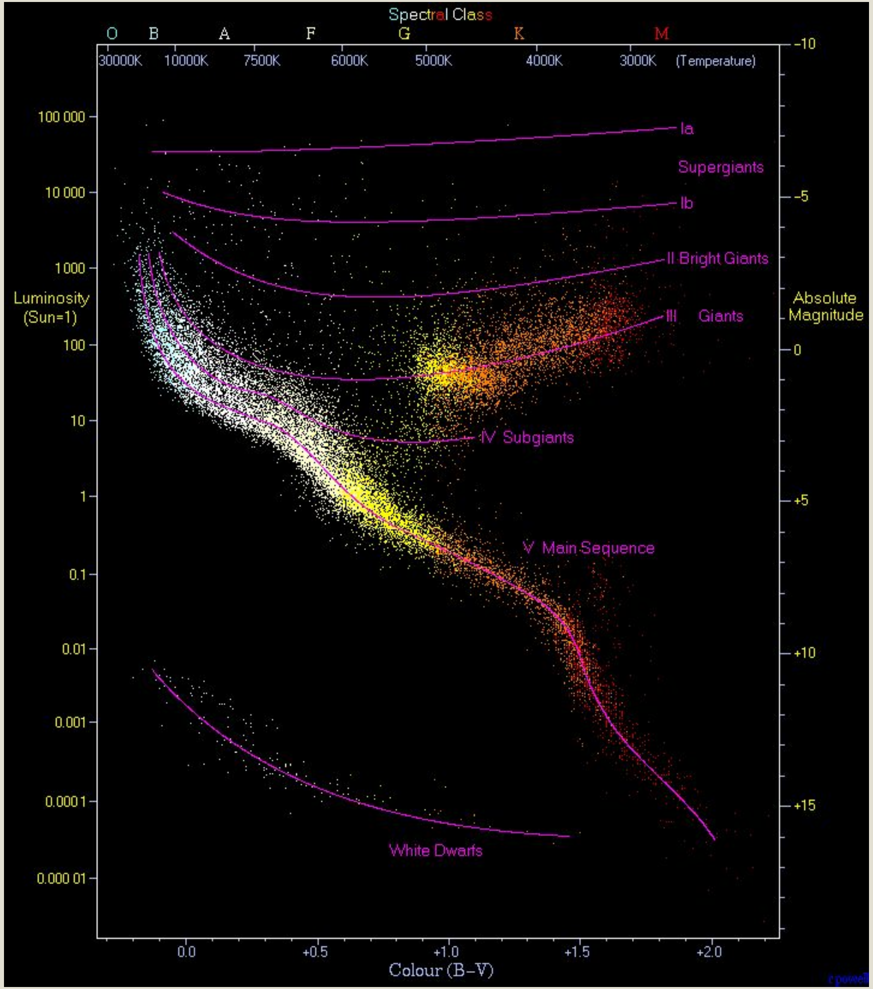
19 DEC 2024

SENIOR PROJECT PROPOSAL

CLASSIFYING STELLAR OBJECTS

AILA, MICHAEL, RAMSES, SIMONA



DATA-DRIVEN ASTRONOMY

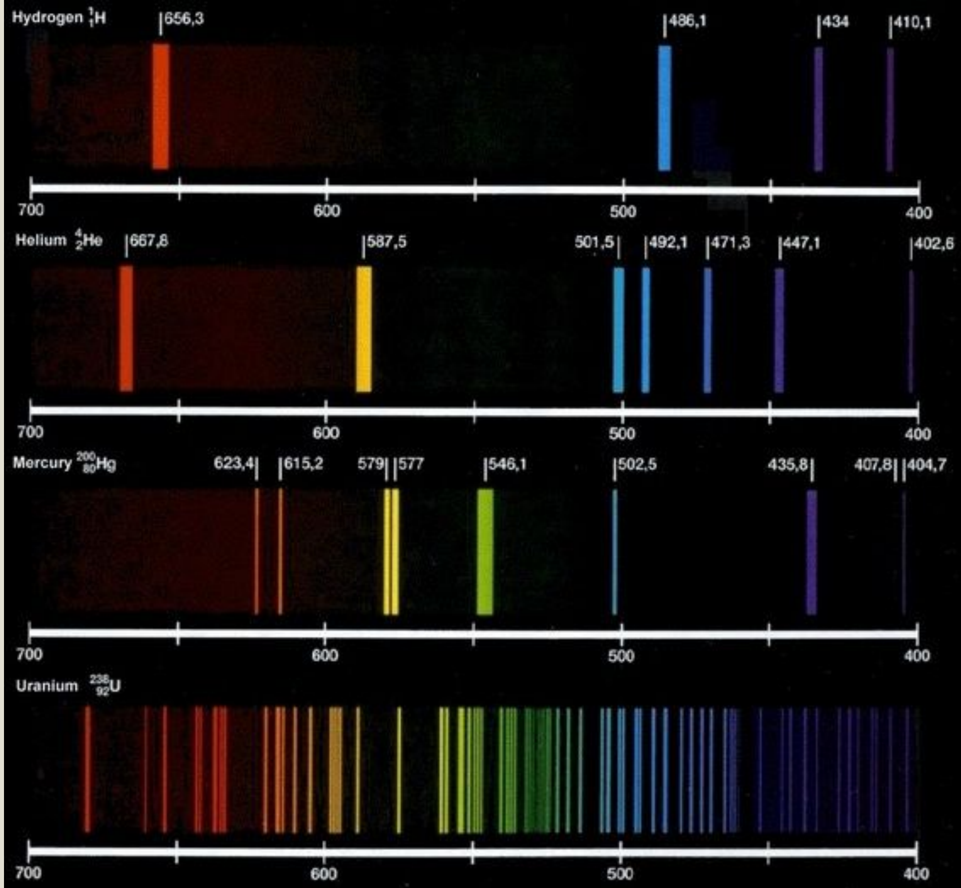
Often called **astroinformatics**, data-driven astronomy is the intersection between data science and the study of astral objects.

Data on hundreds of thousands of stars is collected by several telescopes, both on land and in space, resulting in *petabytes* or even *exabytes* of data that must be processed in order to understand the universe around us.

STELLAR SPECTROSCOPY

One way of analyzing stars is to observe the intensity of different wavelengths of light that are reflected back from the astral object. This is known as **spectroscopy**.

The idea is that different elements that make up the star will absorb and reflect different wavelengths of light. The relative intensities of these wavelengths can give us a rough idea of what stars are made of, and what their temperature is.

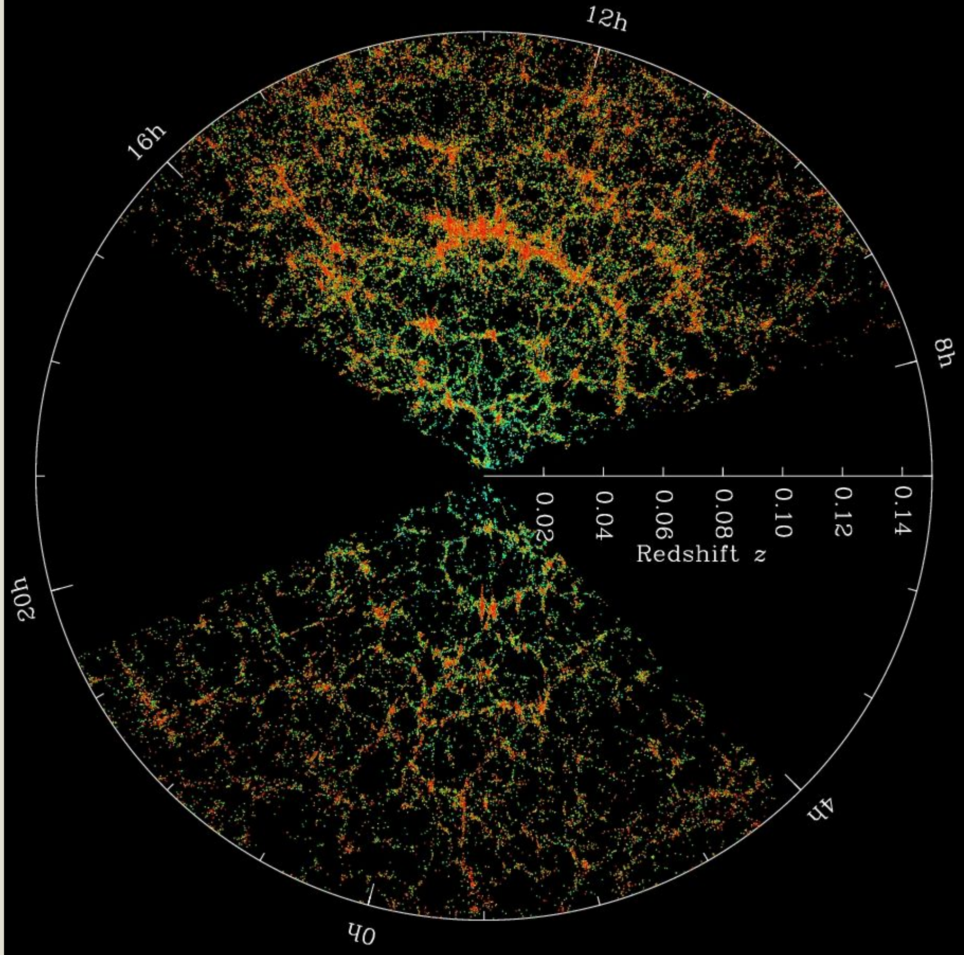


OBTAINING THE DATA

We will be using spectral data from the **Sloan Digital Sky Survey (SDSS)**, which is a project led by a consortium of universities that aims to survey as many stars and galaxies in our universe using multi-spectral imaging.

The result of this surveying are the spectrograms that we saw previously, but for several hundreds of thousands of stars!

The image to the right is a “photo” of the universe as seen by the SDSS, with color representing the density of astronomical bodies in a given location.

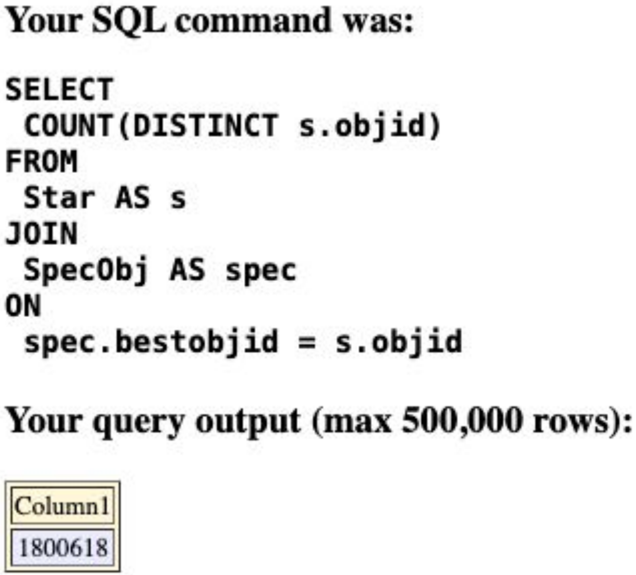


The data for SDSS is accessed through a website called **SkyServer**, which allows us to query data in bulk and for specific stars as well.

While the total dataset has over *260 million* stars, not all of them have the spectrogram data which lays the foundation of our data science project.

To find how many stars we would actually be considering, the following SQL query was executed on SkyServer.

The number came out to **1,800,618 different stars**!



Your SQL command was:

```
SELECT
 COUNT(DISTINCT s.objid)
FROM
 Star AS s
JOIN
 SpecObj AS spec
ON
 spec.bestobjid = s.objid
```

Your query output (max 500,000 rows):

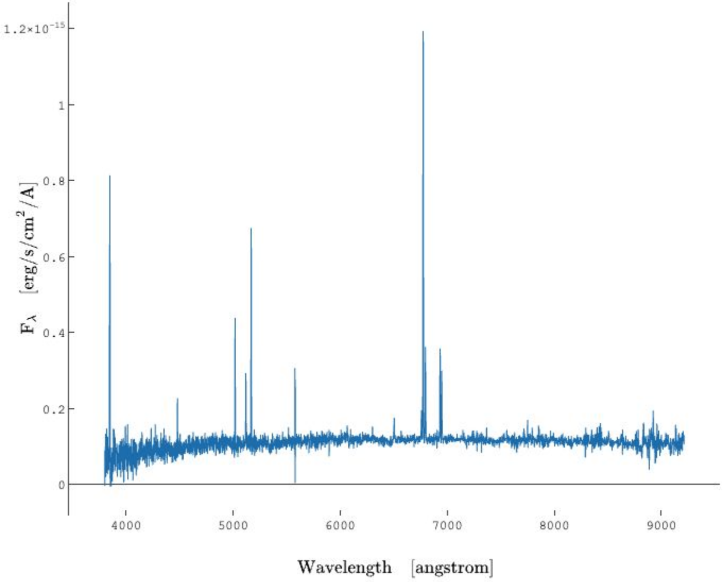
Column1
1800618

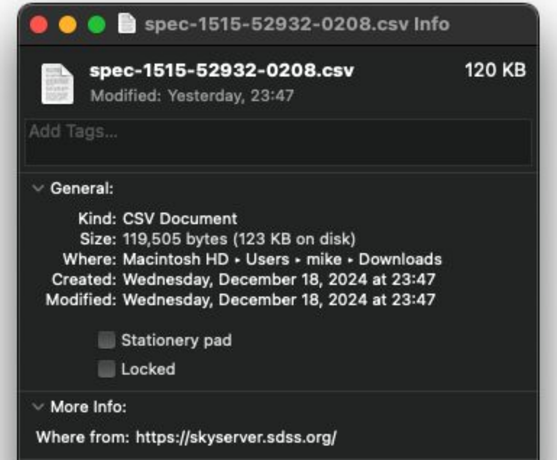
PROCESSING THE DATA

The spectrogram data comes in as a CSV file which contains each wavelength sample and the corresponding "flux", or how much the power is changing from one wavelength to the next.

Each spectrogram, one per star, comes in at roughly 125 KB. The file also contains "best fit" and "sky flux" data which is irrelevant to us, leaving us with a file size of roughly 60 KB when excluded.

With our set of just over 1.8 million stars, this makes our dataset a whopping 110 GB large. This is far more than we can load into the GPU at once, so we will need to chunk the data and process it bit by bit.




spec-1515-52932-0208.csv Info

spec-1515-52932-0208.csv 120 KB
Modified: Yesterday, 23:47

General:

Kind: CSV Document
Size: 119,505 bytes (123 KB on disk)
Where: Macintosh HD ▸ Users ▸ mike ▸ Downloads
Created: Wednesday, December 18, 2024 at 23:47
Modified: Wednesday, December 18, 2024 at 23:47

□ Stationery pad
□ Locked

More Info:

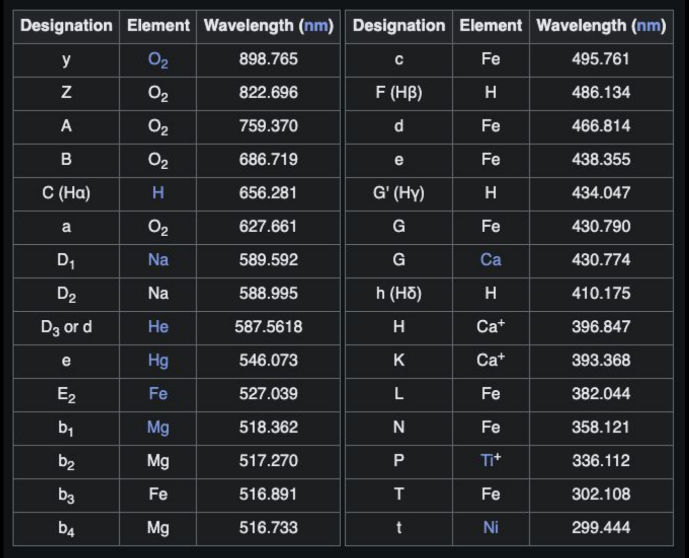
Where from: https://skyserver.sdss.org/

For each star, we will analyze the spectral data and identify which identifying wavelengths are absorbed.

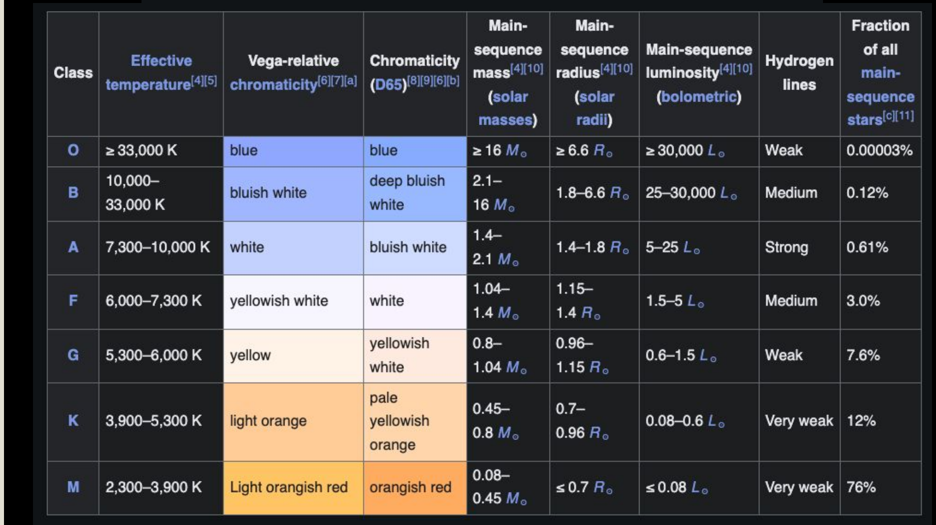
The first table pictured to the right shows various wavelengths and which elements they indicate the presence of. This gives us a very rough **element composition** of the star.

For each star, the temperature can also be estimated by taking the maximum flux seen in the spectrograph. This allows us to determine the **color** of the star when observed and its **class**.

This will be done in parallel to increase throughput and to allow the entire spectrum to be analyzed at once rather than serially.



Designation	Element	Wavelength (nm)	Designation	Element	Wavelength (nm)
y	O_2	898.765	c	Fe	495.761
Z	O_2	822.696	F (Hβ)	H	486.134
A	O_2	759.370	d	Fe	466.814
B	O_2	686.719	e	Fe	438.355
C (Hα)	H	656.281	G' (Hγ)	H	434.047
a	O_2	627.661	G	Fe	430.790
D_1	Na	589.592	G	Ca	430.774
D_2	Na	588.995	h (Hδ)	H	410.175
D_3 or d	He	587.5618	H	Ca^+	396.847
e	Hg	546.073	K	Ca^+	393.368
E_2	Fe	527.039	L	Fe	382.044
b_1	Mg	518.362	N	Fe	358.121
b_2	Mg	517.270	P	Ti^+	336.112
b_3	Fe	516.891	T	Fe	302.108
b_4	Mg	516.733	t	Ni	299.444



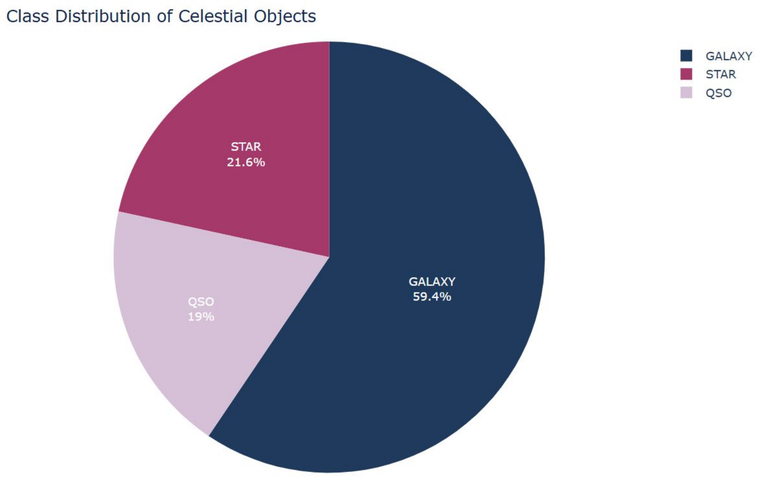
Class	Effective temperature[4][5]	Vega-relative chromaticity[6][7][a]	Chromaticity (D65)[8][9][6][b]	Main-sequence mass[4][10] (solar masses)	Main-sequence radius[4][10] (solar radii)	Main-sequence luminosity[4][10] (bolometric)	Hydrogen lines	Fraction of all main-sequence stars[c][11]
O	≥ 33,000 K	blue	blue	≥ 16 $M_\odot$	≥ 6.6 $R_\odot$	≥ 30,000 $L_\odot$	Weak	0.00003%
B	10,000–33,000 K	bluish white	deep bluish white	2.1–16 $M_\odot$	1.8–6.6 $R_\odot$	25–30,000 $L_\odot$	Medium	0.12%
A	7,300–10,000 K	white	bluish white	1.4–2.1 $M_\odot$	1.4–1.8 $R_\odot$	5–25 $L_\odot$	Strong	0.61%
F	6,000–7,300 K	yellowish white	white	1.04–1.4 $M_\odot$	1.15–1.4 $R_\odot$	1.5–5 $L_\odot$	Medium	3.0%
G	5,300–6,000 K	yellow	yellowish white	0.8–1.04 $M_\odot$	0.96–1.15 $R_\odot$	0.6–1.5 $L_\odot$	Weak	7.6%
K	3,900–5,300 K	light orange	pale yellowish orange	0.45–0.8 $M_\odot$	0.7–0.96 $R_\odot$	0.08–0.6 $L_\odot$	Very weak	12%
M	2,300–3,900 K	Light orangish red	orangish red	0.08–0.45 $M_\odot$	≤ 0.7 $R_\odot$	≤ 0.08 $L_\odot$	Very weak	76%

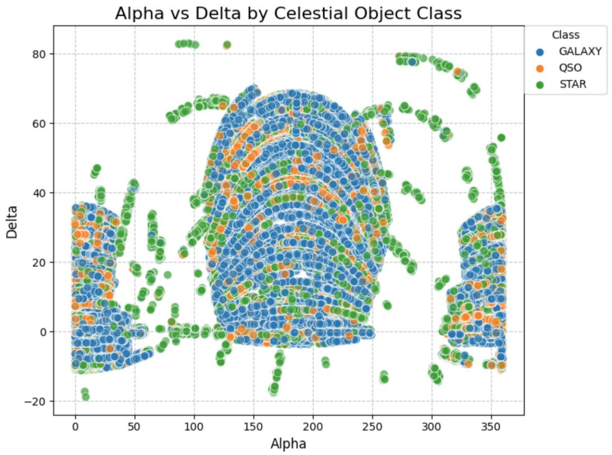
UNDERSTANDING THE DATA

The point of data science is to tell a story through effective imagery. Our story is one of physics.

With our data analysis, we can plot the temperature, colors, and elemental compositions of planets. We could also try to compare this data against the distance (represented by red shift magnitude in astronomy).

While there are many sources that will show you the temperature and size of stars, few will dive into the elements that make them up. We hope to help fill this niche.

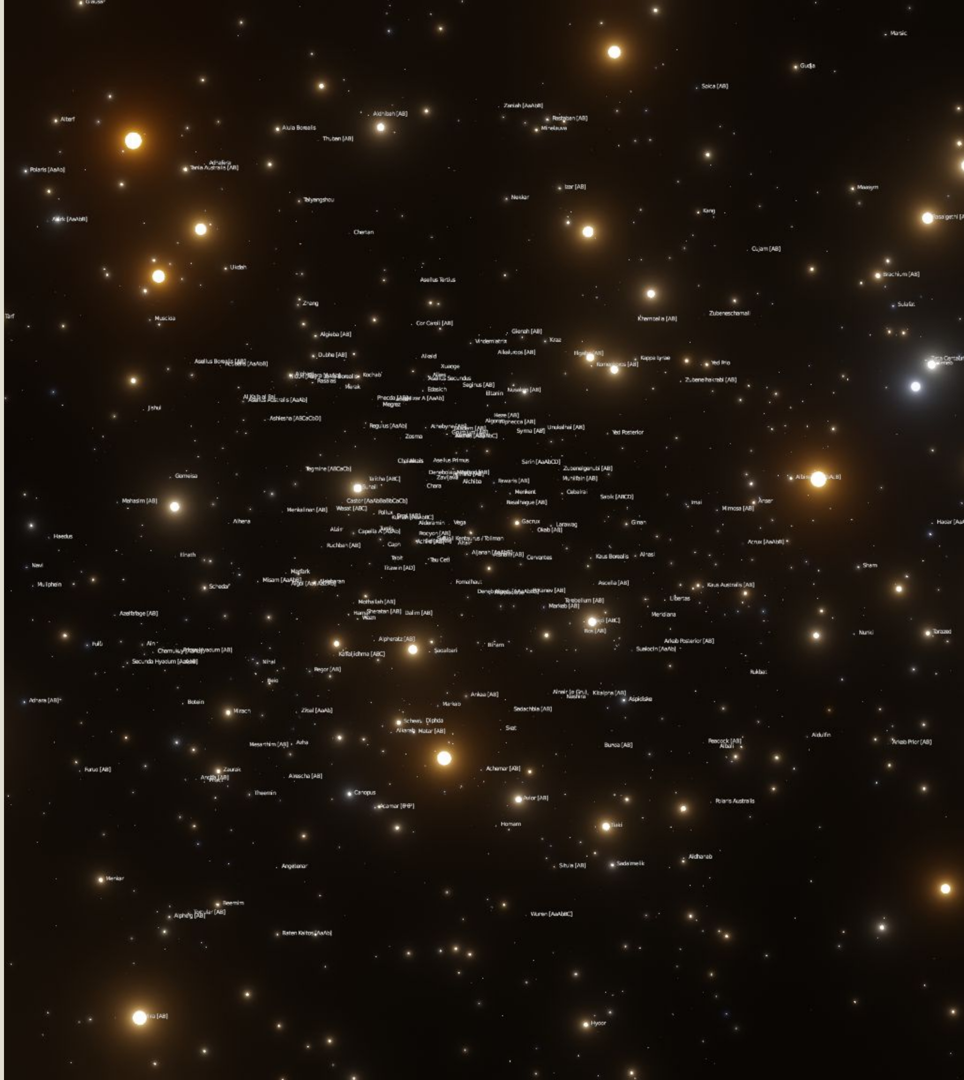




We would also like to create a separate program that uses GPU shaders to draw an interactive star map (similar to the one shown to the right).

A shader is simply a kernel that returns a color for each pixel on some screen, and it’s how graphics cards render… graphics!

In our star map, you would see the colors of the stars we computed before, but you would also be able to click on individual stars to see their elemental composition.



POTENTIAL EXTENSION TO MACHINE LEARNING

Clustering Similar Stars: Use unsupervised learning (e.g., K-Means, DBSCAN) to identify groups of stars with similar compositions or properties.

Anomaly Detection: Detect unusual stars or potential outliers, such as candidates for rare star types, using anomaly detection methods (e.g., Isolation Forests).

Classification Models: By using classification models, we can achieve high accuracy in identifying different types of celestial objects, aiding in more precise scientific research.

Machine Learning Models for Star Classification

- **Logistic Regression:** Effective for binary classification tasks, can be extended for multi-class problems like spectral classification.
- **K-Nearest Neighbors (KNN):** Useful for identifying similar stars based on features like temperature and luminosity; sensitive to the choice of n_neighbors.
- **Decision Tree:** Intuitive model that helps classify stars based on sequential splits of features.
- **Gaussian Naive Bayes (GaussianNB):** Assumes feature independence and works well with probabilistic star classifications.
- **Random Forest:** Combines multiple decision trees for robust classification and better handling of overfitting.
- **XGBoost:** Gradient boosting algorithm offering high accuracy and efficiency for larger datasets.

Project Framework

1. Data Acquisition

- Source: Sloan Digital Sky Survey (SDSS)
- Tools: SQL (SkyServer)

2. Data Preprocessing

- Tools: Python (Pandas), SQL
- Tasks: Cleaning data, normalizing data, preparing chunks for GPU processing

3. GPU-Based Processing

- Framework: CUDA, RAPIDS (cuDF, cuGraph, cuML)
- Tasks: Parallel data processing, feature extraction from spectrograms, computing properties like temperature

4. Machine Learning Models

- Framework: TensorFlow & Scikit-learn
- Tasks: Clustering, anomaly detection, classification

5. Visualization

- Framework: RAPIDS visualization libraries
- Tools: Dash for interactive exploration of star maps.

Software Tools and Packages

CUDA: For GPU parallel-processing. It will allow us to process large datasets efficiently by dividing tasks into smaller chunks that can be executed concurrently.

Pandas: Enables efficient data manipulation and preprocessing, such as cleaning and normalization of spectral data.

Scikit-learn: For clustering and anomaly detection of star data.

TensorFlow: To build classification models that categorize celestial objects based on their spectral data.

RAPIDS cuGraph: GPU-accelerated libraries that facilitate large-scale data visualization and clustering tasks.

SQL: To query and filter the SDSS data for relevant spectrograms and properties.

GitHub: For team collaboration.

Groups’ Members Guidelines

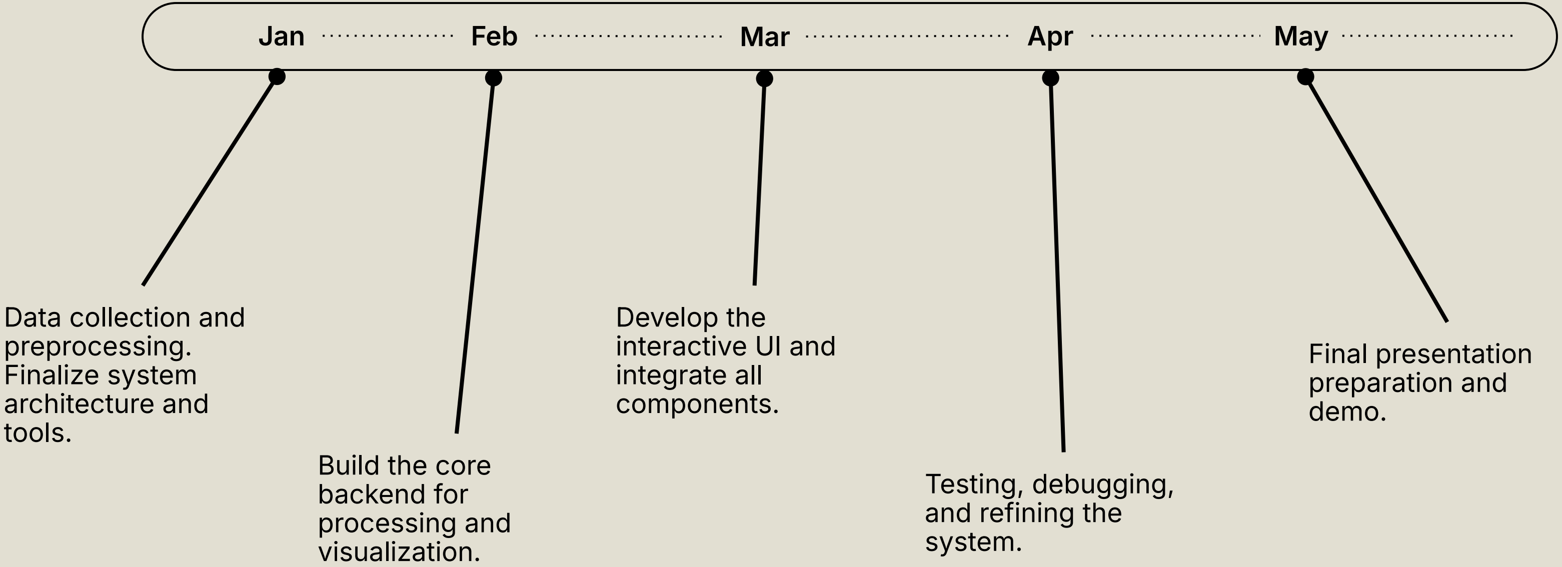
The set of guidelines that all group member must adhere to:

- Our work should be divided evenly (until we can formulate proper group roles)
- We must attend meetings
- If a group member is facing complications in IRL, they should let the others know immediately
- At least one of us should have a working build of the final product
- Use github to share the progress of our work

Groups’ Members Roles

Aila (Front-End/UI Developer): Responsible for building the interactive UI, integrating visualization libraries
Michael (Back-End/Data Specialist): Handling data preprocessing, data cleaning, and CUDA implementation for parallel data processing.
Ramses (Data Scientist/Analyst): Working on the interpretation of star data, helping with the mapping of temperature to color, and ensuring the accuracy of star composition data.
Simona (Machine Learning Specialist): Responsible for leveraging machine learning techniques to analyze and classify stars based on spectral data.

Timeline



Jan

Data collection and preprocessing. Finalize system architecture and tools.

Feb

Build the core backend for processing and visualization.

Mar

Develop the interactive UI and integrate all components.

Apr

Testing, debugging, and refining the system.

May

Final presentation preparation and demo.

QUESTIONS?