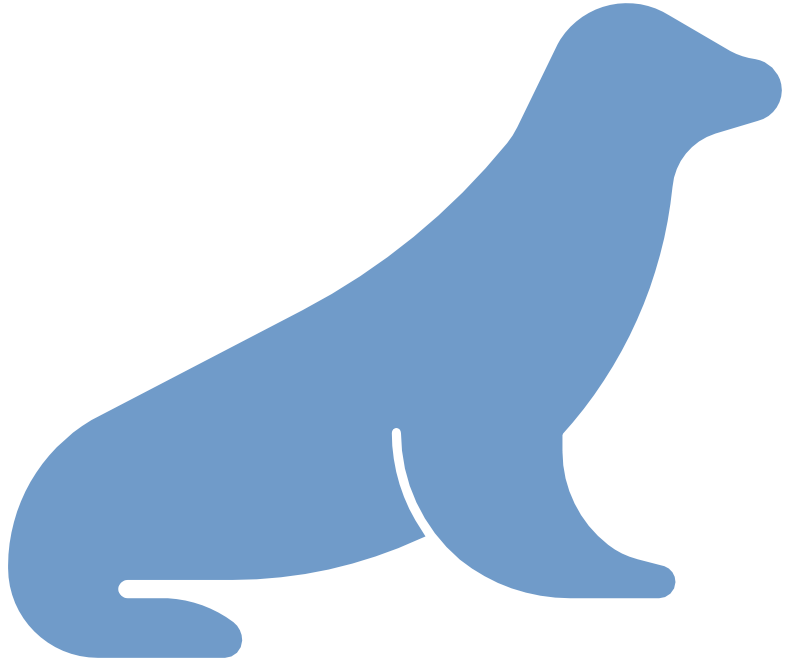
Internship Project Presentation

Devesh Vijaywargiya (t-devija)

Motivation

The core motivation for this project was to make the Heron portal more user friendly by improving its user experience.



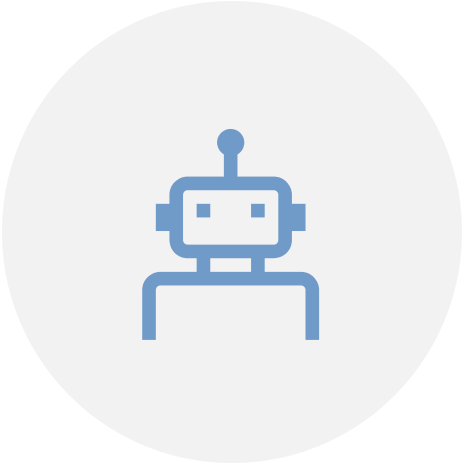
Project Idea

This project aims at improving the user experience of the Heron portal by adding user level and dataset level metrics as a part of the user interface. Those using the portal will get a better understanding of all the datasets being offered, thereby helping them make a better decision about their Dataset requirement.

Metrics

- **User Level**
  - Number of requests
  - Successful transactions
  - Reliability
  - Favorite Datasets
- **Dataset Level**
  - Extraction Cost
  - Number of requests & successful requests
  - User completeness parameter

Tech Stack

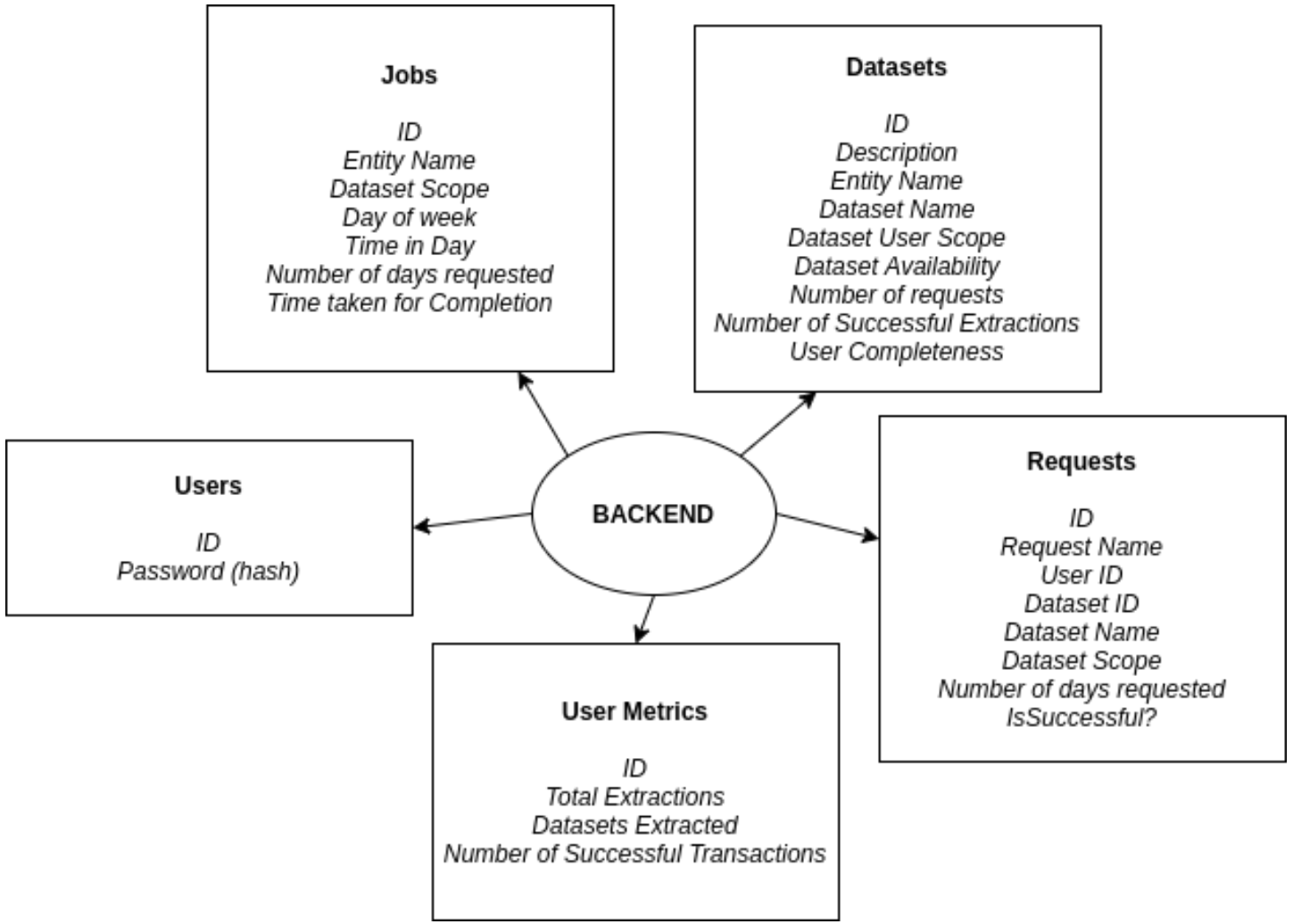


FLASK FRAMEWORK (BASED ON PYTHON) TO BUILD THE APIS AND SERVICES

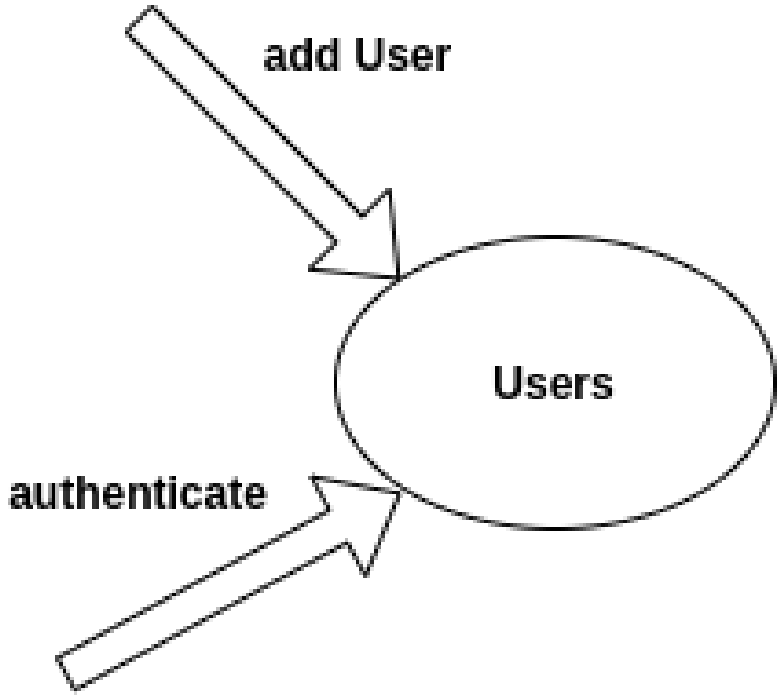
REACTJS FRAMEWORK/LIBRARY TO CREATE THE FRONTEND DEMONSTRATING THE APIS

DESIGN

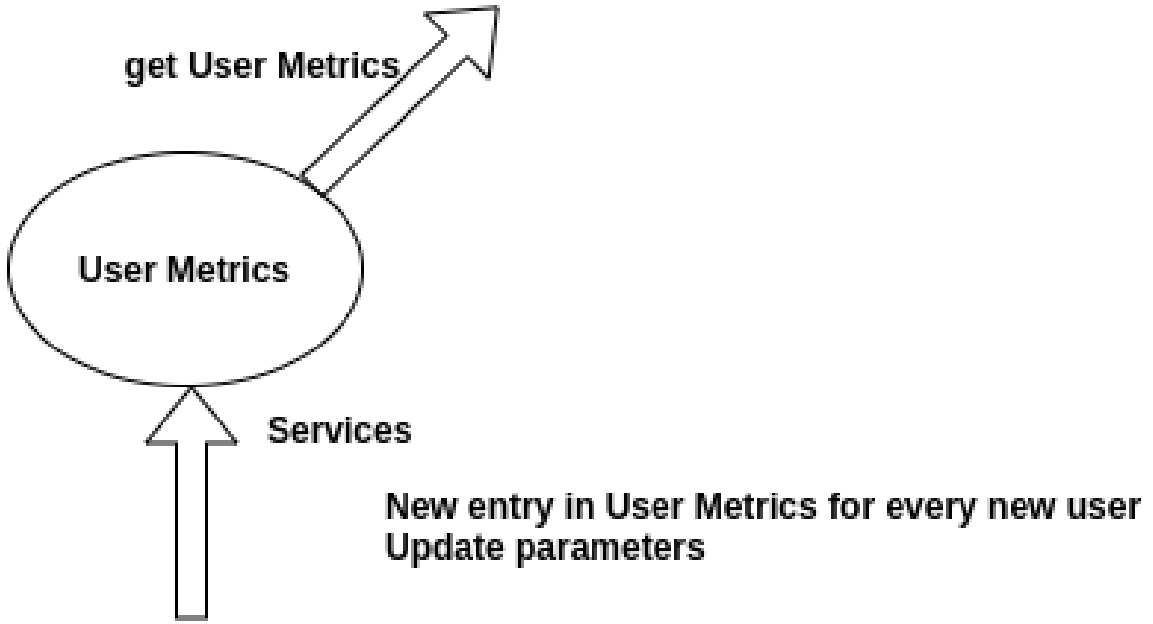
Containers


Jobs
ID
Entity Name
Dataset Scope
Day of week
Time in Day
Number of days requested
Time taken for Completion
Datasets
ID
Description
Entity Name
Dataset Name
Dataset User Scope
Dataset Availability
Number of requests
Number of Successful Extractions
User Completeness
Users
ID
Password (hash)
BACKEND
Requests
ID
Request Name
User ID
Dataset ID
Dataset Name
Dataset Scope
Number of days requested
IsSuccessful?
User Metrics
ID
Total Extractions
Datasets Extracted
Number of Successful Transactions

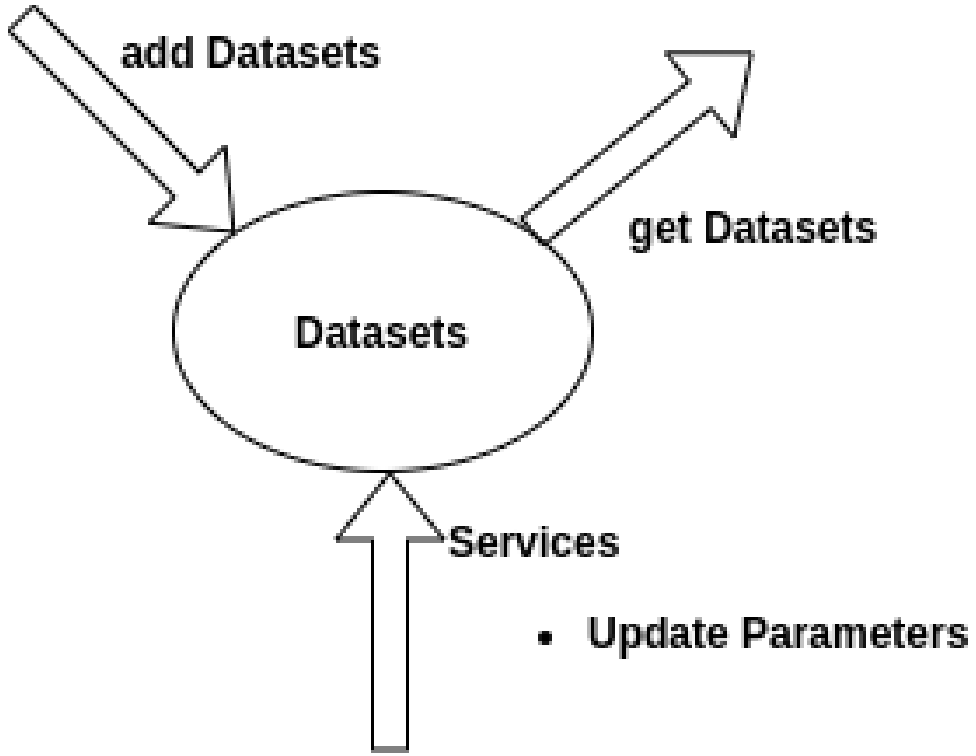
APIs and Services



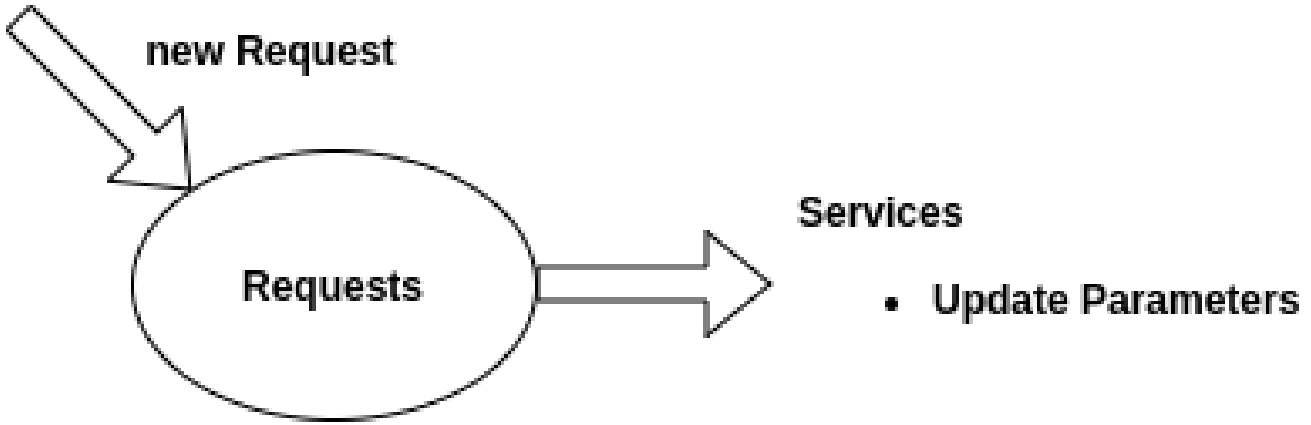
APIs and Services



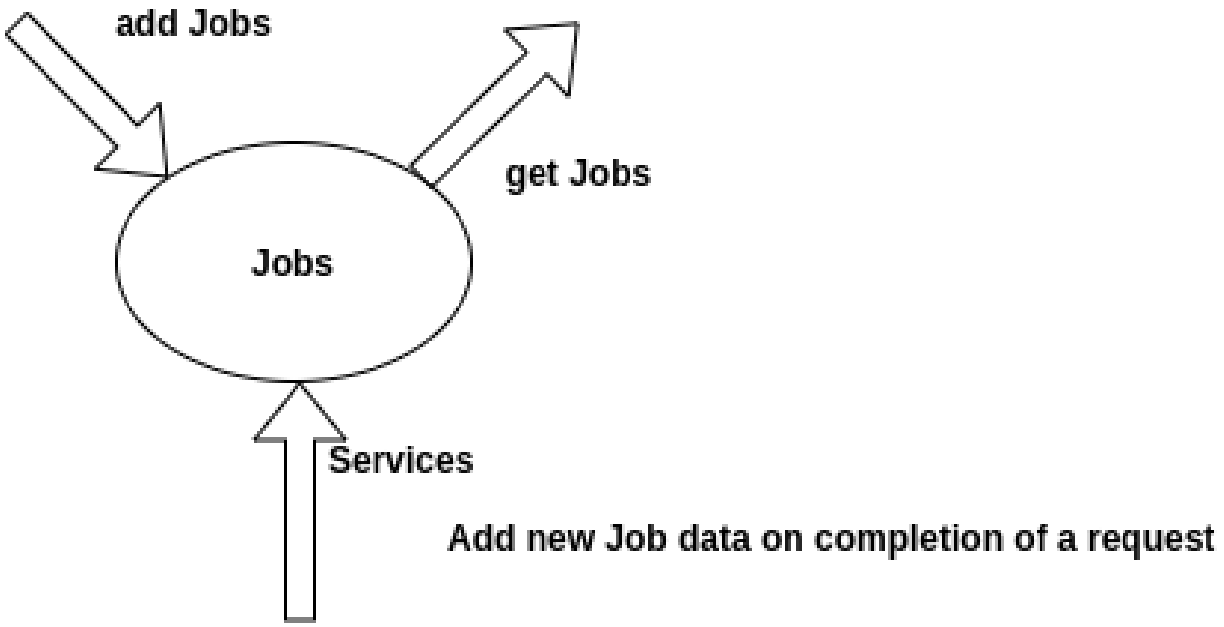
APIs and Services

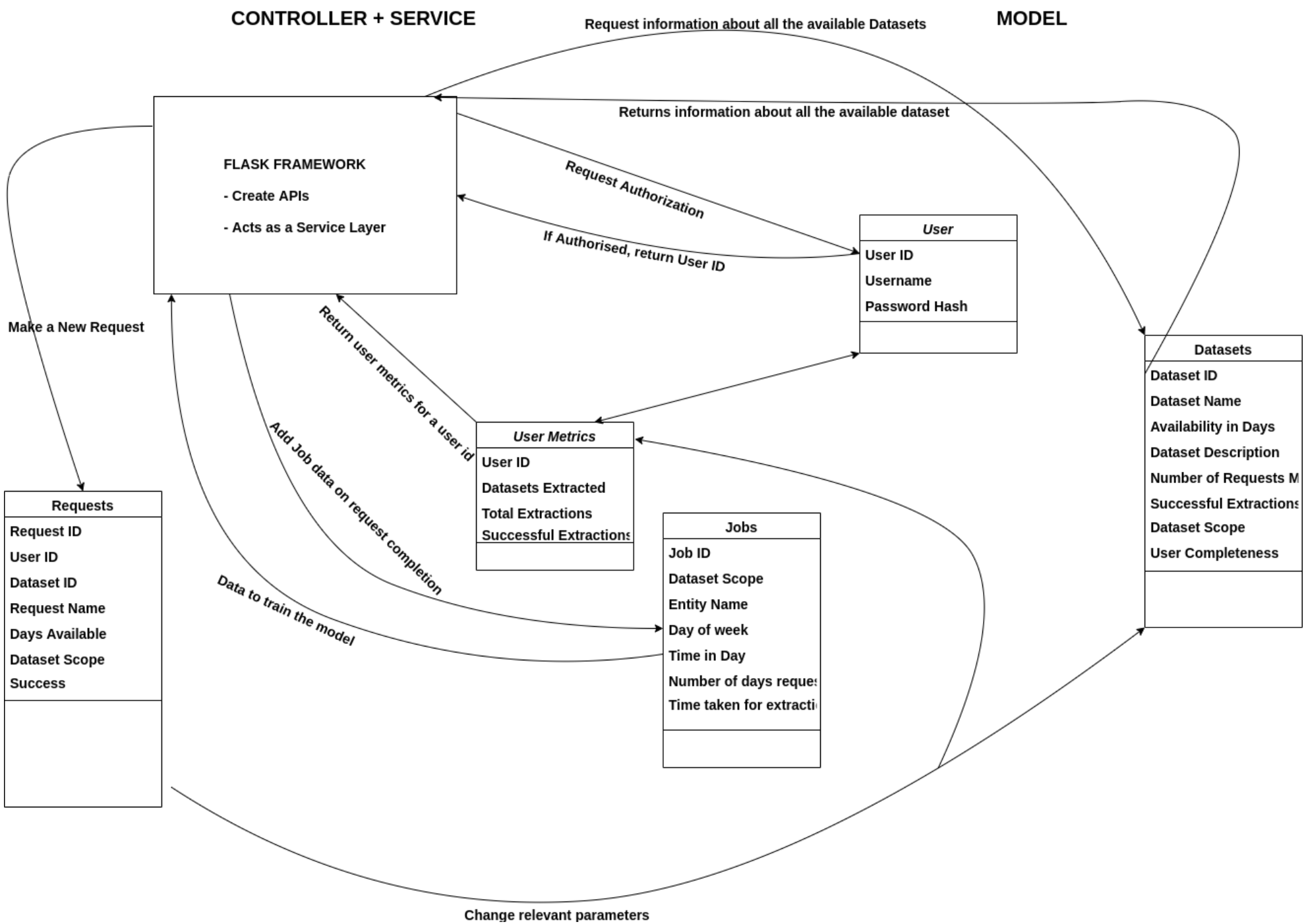


APIs and Services

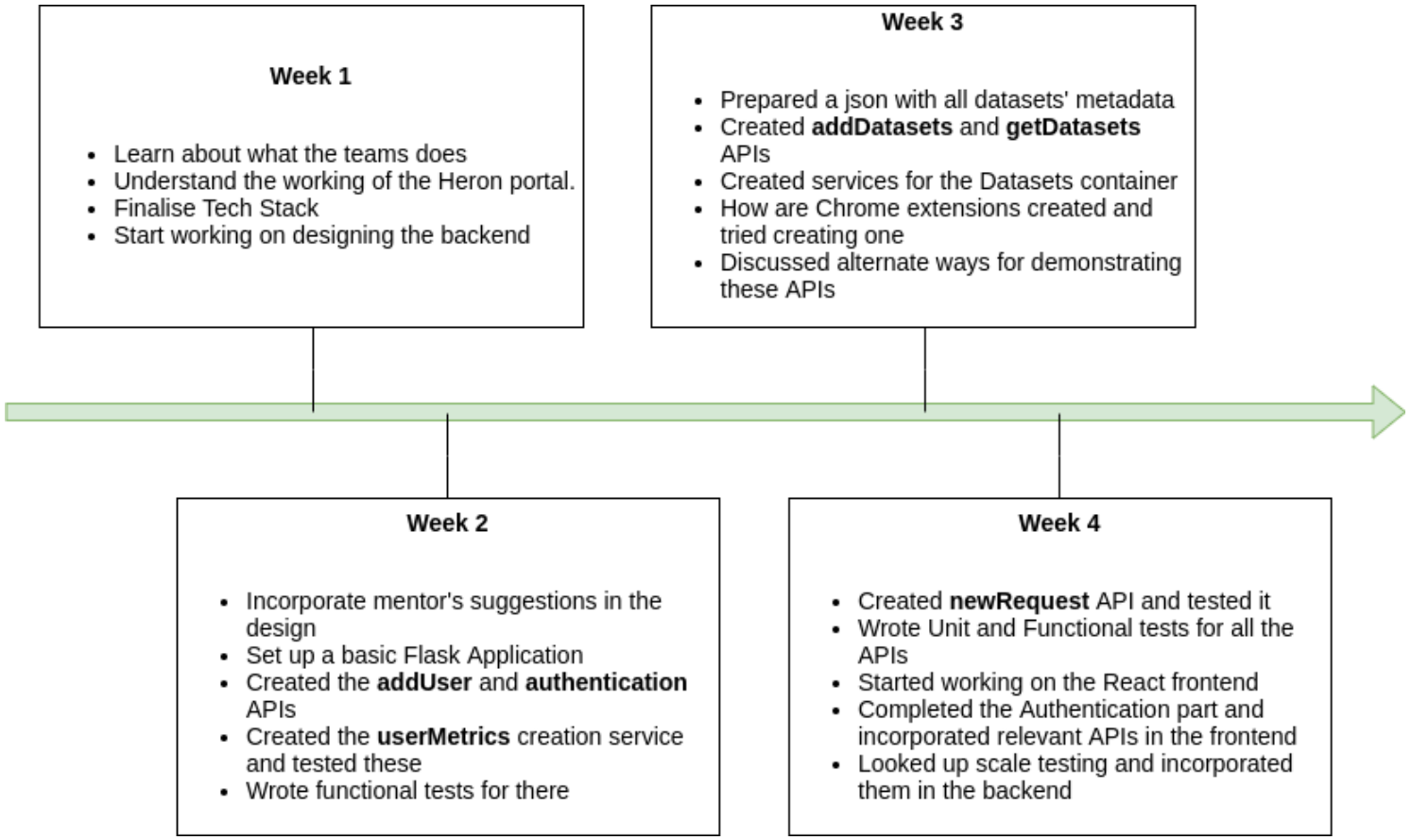


APIs and Services

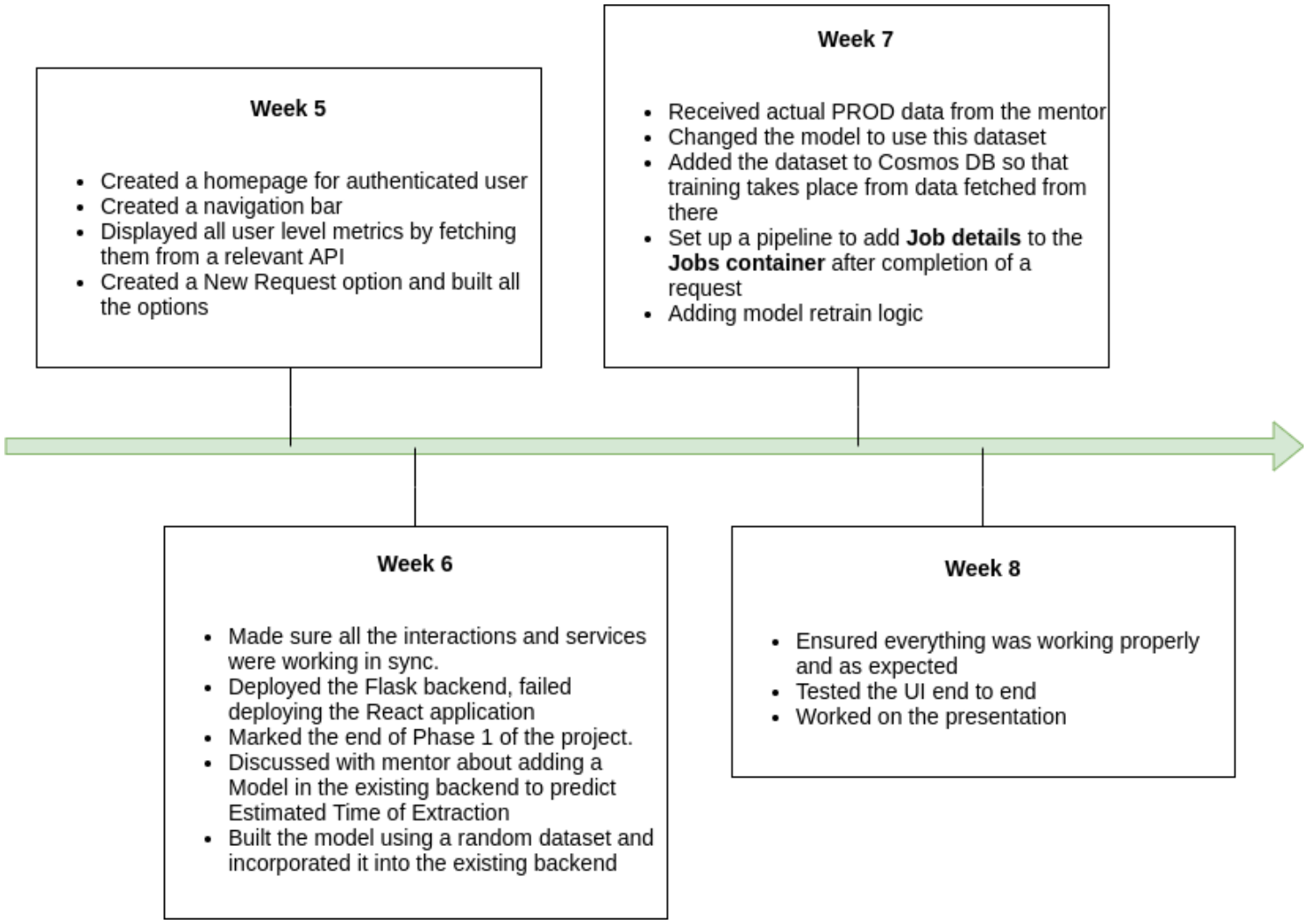



CONTROLLER + SERVICE
MODEL
Request information about all the available Datasets
Returns information about all the available dataset
FLASK FRAMEWORK
- Create APIs
- Acts as a Service Layer
Request Authorization
If Authorised, return User ID
Make a New Request
Return user metrics for a user id
Add Job data on request completion
Data to train the model
Change relevant parameters
User
User ID
Username
Password Hash
Datasets
Dataset ID
Dataset Name
Availability in Days
Dataset Description
Number of Requests M
Successful Extractions
Dataset Scope
User Completeness
User Metrics
User ID
Datasets Extracted
Total Extractions
Successful Extractions
Requests
Request ID
User ID
Dataset ID
Request Name
Days Available
Dataset Scope
Success
Jobs
Job ID
Dataset Scope
Entity Name
Day of week
Time in Day
Number of days reque
Time taken for extracti

Timeline (1)


Week 1
Learn about what the teams does
Understand the working of the Heron portal.
Finalise Tech Stack
Start working on designing the backend
Week 2
Incorporate mentor's suggestions in the design
Set up a basic Flask Application
Created the addUser and authentication APIs
Created the userMetrics creation service and tested these
Wrote functional tests for there
Week 3
Prepared a json with all datasets' metadata
Created addDatasets and getDatasets APIs
Created services for the Datasets container
How are Chrome extensions created and tried creating one
Discussed alternate ways for demonstrating these APIs
Week 4
Created newRequest API and tested it
Wrote Unit and Functional tests for all the APIs
Started working on the React frontend
Completed the Authentication part and incorporated relevant APIs in the frontend
Looked up scale testing and incorporated them in the backend

Timeline (2)


Week 5
Created a homepage for authenticated user
Created a navigation bar
Displayed all user level metrics by fetching them from a relevant API
Created a New Request option and built all the options
Week 7
Received actual PROD data from the mentor
Changed the model to use this dataset
Added the dataset to Cosmos DB so that training takes place from data fetched from there
Set up a pipeline to add Job details to the Jobs container after completion of a request
Adding model retrain logic
Week 6
Made sure all the interactions and services were working in sync.
Deployed the Flask backend, failed deploying the React application
Marked the end of Phase 1 of the project.
Discussed with mentor about adding a Model in the existing backend to predict Estimated Time of Extraction
Built the model using a random dataset and incorporated it into the existing backend
Week 8
Ensured everything was working properly and as expected
Tested the UI end to end
Worked on the presentation

Key Learnings

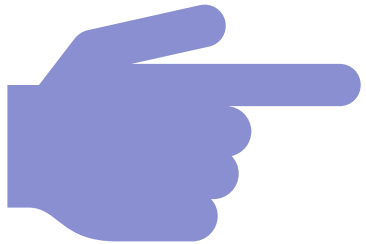
- Working of the Heron portal
- NoSQL databases, their types | Analysis | Which is better for us?
- Cosmos DB Python SDK
- Synchronous | Asynchronous | Single Threaded | Multi-threaded programming
- Flask vs NodeJS (Connection pooling)
- Building Chrome Extensions
- Unit | Functional | Smoke | Scale testing
- Configuring React apps for deployment
- Deploying applications on Azure
- Creating models and tweaking them to achieve better results

Challenges

- Understanding the heron portal pipeline
- Chrome extension debacle
- Deployment issues
- Flask directory structure
- Testing in Flask
- Figuring out NoSQL databases and Cosmos DB

Further Development

The model can be improved by adding more features



The UI created can be developed further and designed in a way more integratable with the actual portal

DEMO