IOWA STATE UNIVERSITY

Detecting and Predicting Clusters of Evolving Binary Stars

sdmay21-30

Adam Corpstein, Joel Holm, Willis Knox, Becker Mathie, Philip Payne, Ethan Vander Wiel

Advisor/Client: Goce Trajcevski

Overview

- Project Background
- Implementation
- Demo
- Lessons Learned

Project Background

Problem Statement

Problem

- Large amount of binary star data
- Hard to determine which systems are similar
- Want to find/determine evolution patterns

Solution

- App to track stellar evolution
- Cluster binary star systems
- Aimed at astrophysicists
- Cluster formation detection over time

Requirements

Functional

- Web-based
- Accept User Input
  - Clustering Method
  - Attributes of Interest
  - Attribute Weights
  - Time Intervals
- Display Resulting Graphs

Non-Functional

- Requests placed into queue
- 24/7 Uptime
- User-Graph Interaction
- Reasonable Response Time

Requirements cont.

Environmental

- Internet connection
- Remote server connection with database

Economic

- User needs personal computer
- Server space (paid for by ISU)

Intended Users and Uses

- Astrophysicists
- Researchers using stellar data
  - Sloan Digital Sky Survey
  - Gaia Archive

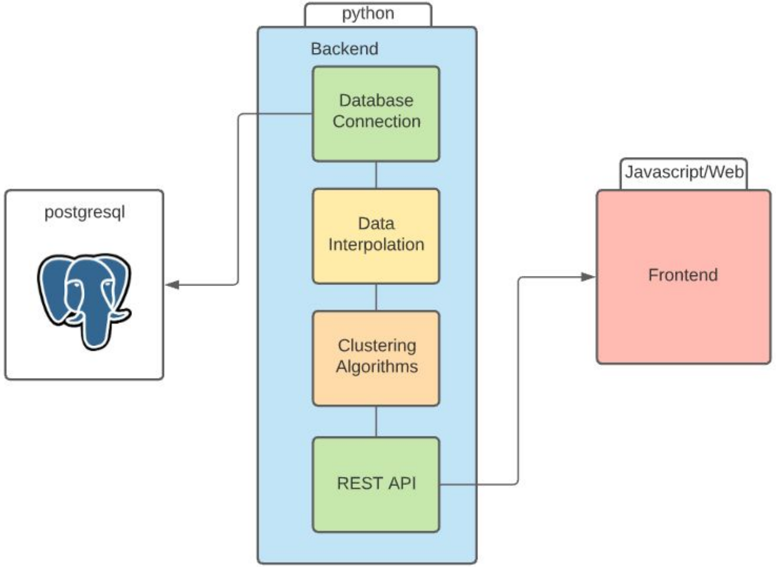
Assumptions & Limitations

- Assumptions
  - Available Internet Access
  - Background Knowledge on data
- Limitations
  - Features in database

Implementation

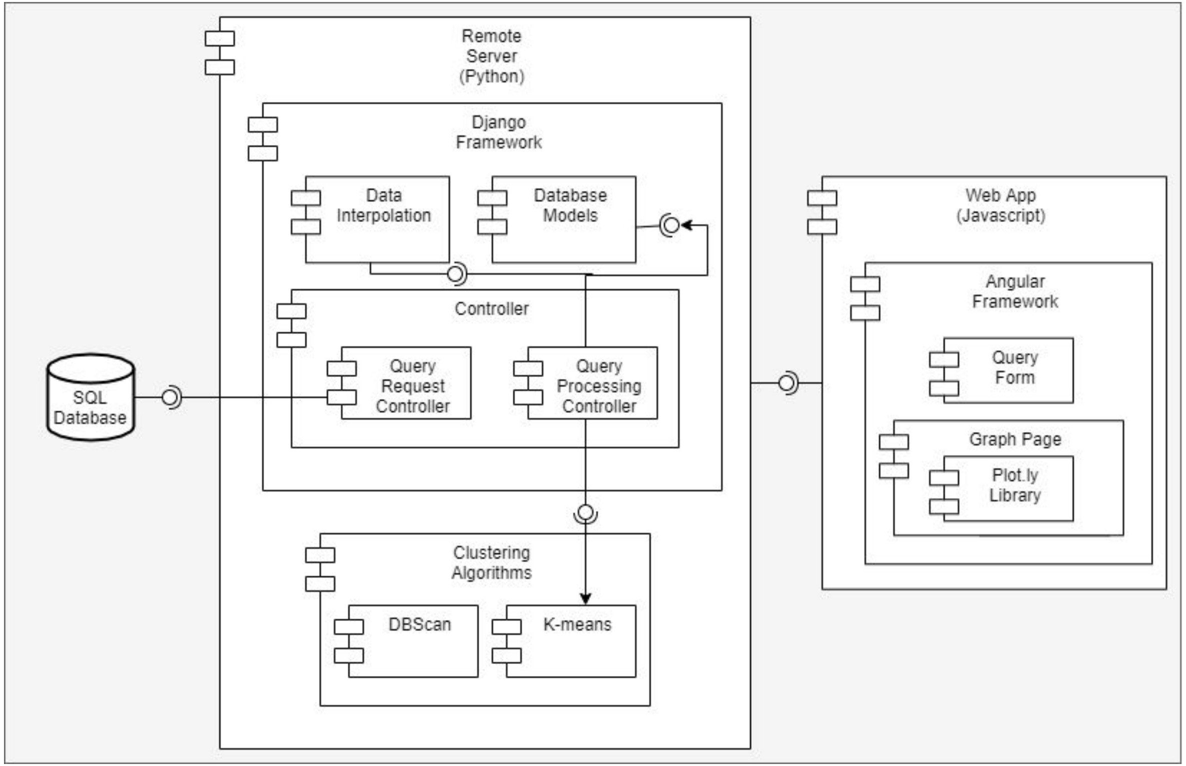
Design Approach

High Level Architecture



Design Approach (cont.)

Component Diagram

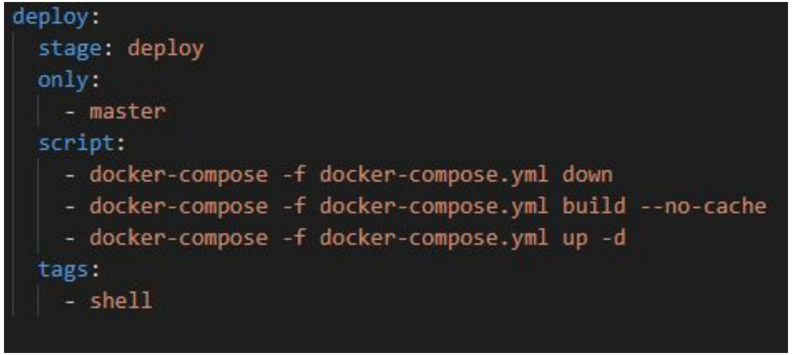


Technologies Used

- Angular Framework (Web)
    - Angular Material
    - Plotly
- Python (API)
    - Django Framework
    - scikit-learn
- PostgreSQL
- Gitlab (CI/CD)
- Docker (For deployment and testing)

CI/CD

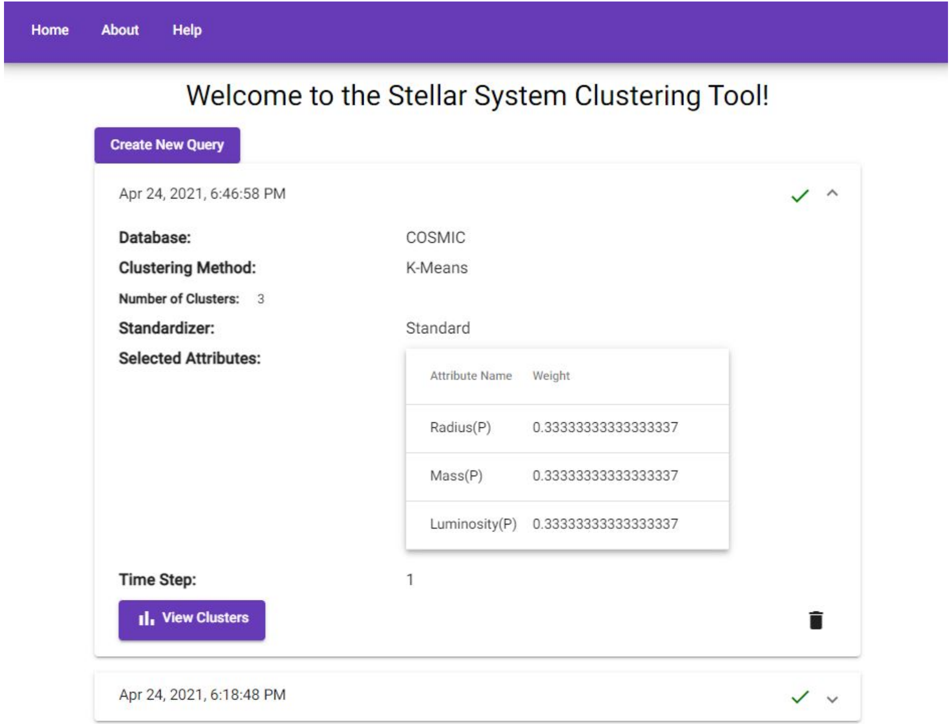
- GitLab CI for pipeline
  - Build, Test, and Deploy stages
  - Only deploy on master
- Deployment
  - Docker image for both UI and API
    - NGINX based image for UI
    - Python based image for API
    - Same as testing environment
  - Docker compose for deployments
    - Easy to deploy anywhere



```
deploy:
  stage: deploy
  only:
    - master
  script:
    - docker-compose -f docker-compose.yml down
    - docker-compose -f docker-compose.yml build --no-cache
    - docker-compose -f docker-compose.yml up -d
  tags:
    - shell
```

User Interface (Home page)

- Create new queries
- View saved queries



Welcome to the Stellar System Clustering Tool!

Apr 24, 2021, 6:46:58 PM ✓

Database: COSMIC

Clustering Method: K-Means

Number of Clusters: 3

Standardizer: Standard

Selected Attributes:

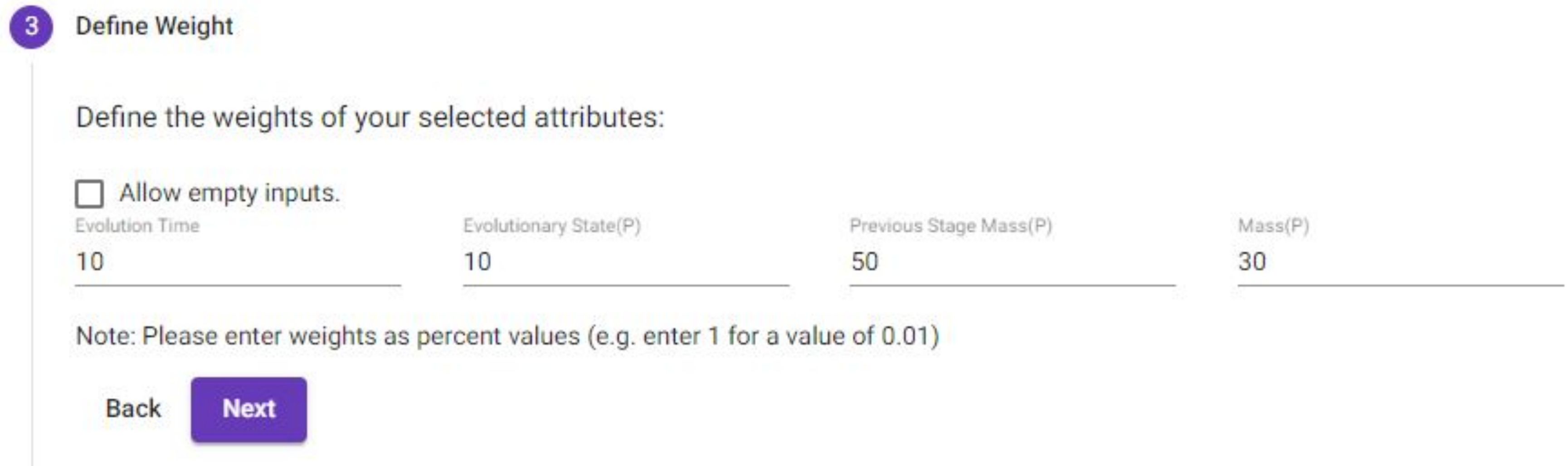
Attribute Name	Weight
Radius(P)	0.33333333333333337
Mass(P)	0.33333333333333337
Luminosity(P)	0.33333333333333337

Time Step: 1

Apr 24, 2021, 6:18:48 PM ✓

User Interface (Query Form Page)

- Database, attribute, weight, cluster algorithm, and extra parameter selection



3 Define Weight

Define the weights of your selected attributes:

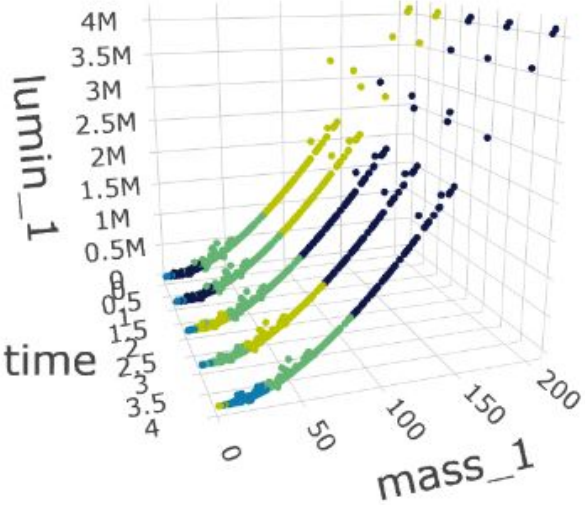
☐ Allow empty inputs.

Evolution Time	Evolutionary State(P)	Previous Stage Mass(P)	Mass(P)
10	10	50	30

Note: Please enter weights as percent values (e.g. enter 1 for a value of 0.01)

User Interface (Graph Page)

- Different graphs
  - 1 attribute over time (2D)
  - 2 attributes over time (3D)
  - 3 attributes with time as input (3D)

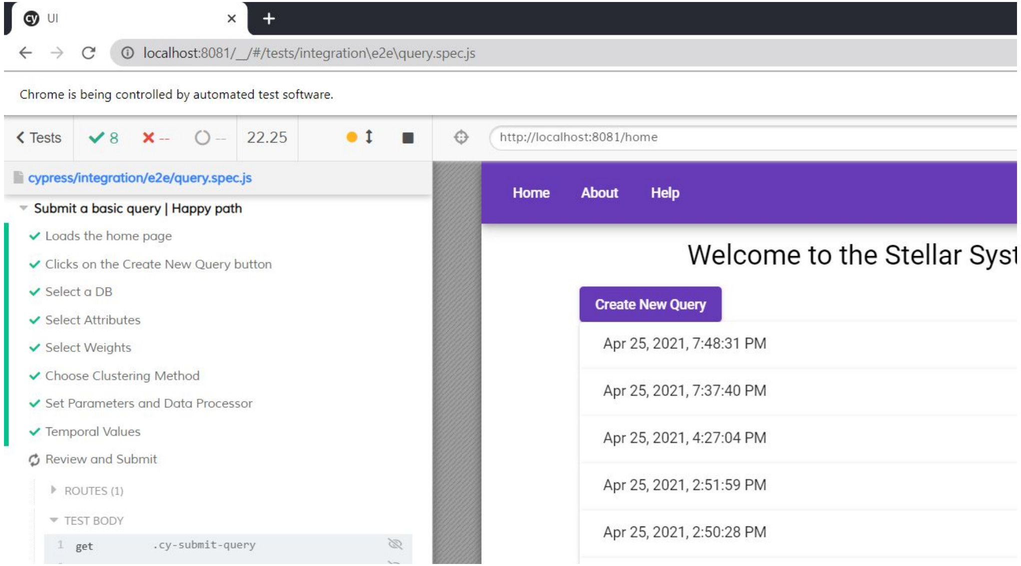


Standards Used

- **Modules For Experiments In Stellar Astrophysics (MESA): Planets, Oscillations, Rotation, And Massive Stars**
  - Standardizes MESAstar, a stellar evolution module
  - Output from these simulations are input into our application

Testing (Frontend)

- Cypress (automated testing tool)
    - End-to-end tests
        - Use Case Scenarios
        - Robustness testing



UI

localhost:8081/__/#/tests/integration\e2e\query.spec.js

Chrome is being controlled by automated test software.

Tests ✓ 8 ✗ -- ○ -- 22.25

http://localhost:8081/home

cypress/integration/e2e/query.spec.js

Submit a basic query | Happy path

- ✓ Loads the home page
- ✓ Clicks on the Create New Query button
- ✓ Select a DB
- ✓ Select Attributes
- ✓ Select Weights
- ✓ Choose Clustering Method
- ✓ Set Parameters and Data Processor
- ✓ Temporal Values
- Review and Submit

ROUTES (1)

TEST BODY

```
get        .cy-submit-query
```

Home About Help

Welcome to the Stellar Syst

Create New Query

Apr 25, 2021, 7:48:31 PM

Apr 25, 2021, 7:37:40 PM

Apr 25, 2021, 4:27:04 PM

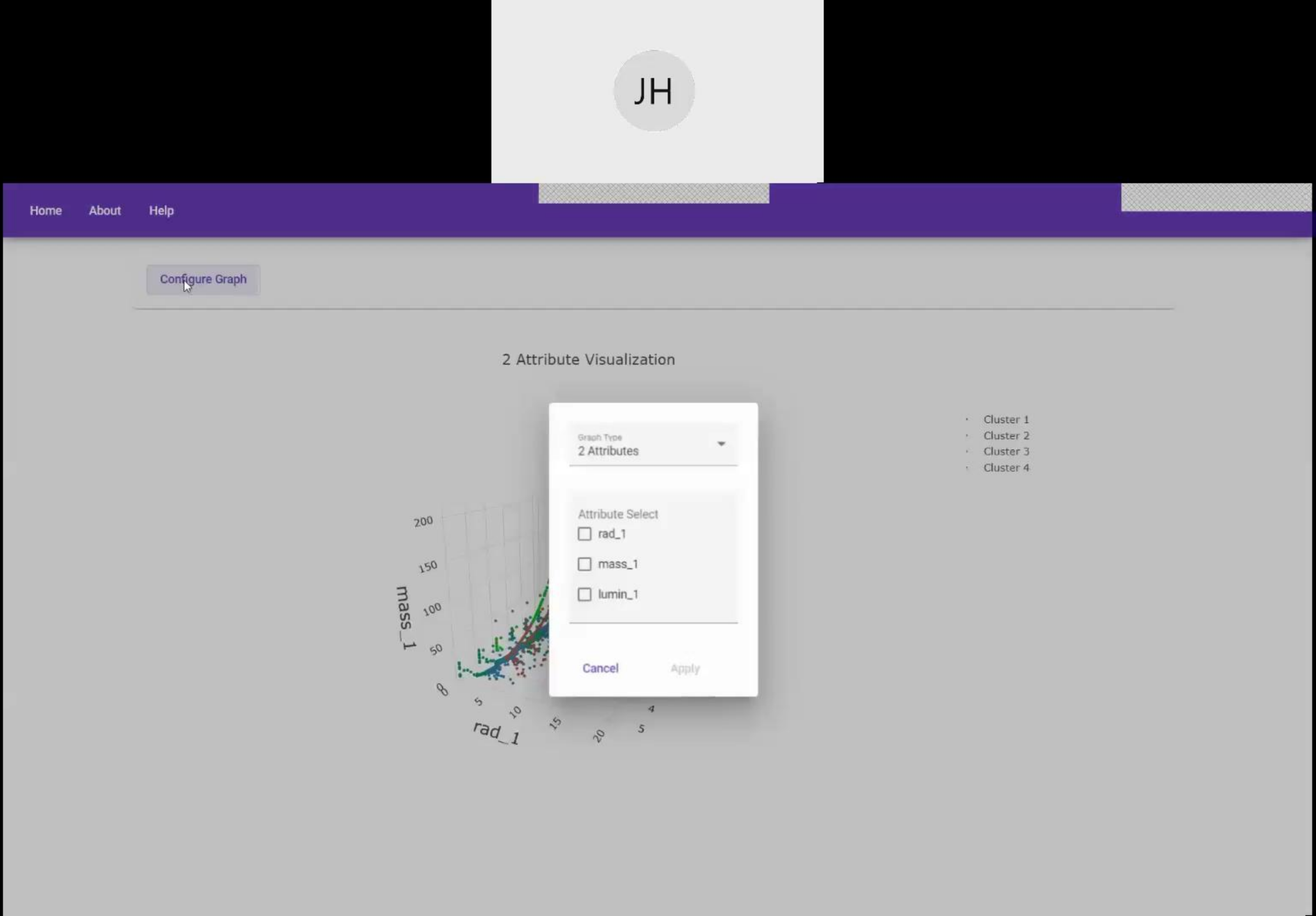
Apr 25, 2021, 2:51:59 PM

Apr 25, 2021, 2:50:28 PM

Testing (Backend)

- Django included testing library
    - Unit testing
        - Clustering algorithms interaction
        - Database access
    - HTTP Request testing
        - Correct handling of requests & responses

Demo



Graph Type
2 Attributes

Attribute Select

☐ rad_1

☐ mass_1

☐ lumin_1

Lessons Learned

Lessons Learned

- Anticipate issues and ask client early on
- Use generality to future proof additions
- Maintain open communication between frontend and backend
- Cleary define testing conditions

Q&A

Email: sdmay21-30@iastate.edu