# HEBSE
## Holistic Exploration of Binary Stellar Evolutions

SDMay25-20

Members: Andrew Snyder, Alex Polston, Alek Norris, Eamon Collins, James Byrd, Svyatoslav Varnitskyy

Client & Advisor: Dr. Goce Trajcevski

# Context

Binary Stars: A pair of stars bound to each other by gravity, revolving around a common center of mass within each others' orbit

The POSYDON research project simulates binary star system evolution, generating massive amounts of data

# Problem Statement

**Problem** - Currently there is no system to effectively search the vast simulation data for particular subsets, i.e. a “complex query.”

**Solution** - Implement a software tool with an intuitive user interface, database setup, complex querying, and data retrieval capabilities with optional OpenAI natural language processing assistance.

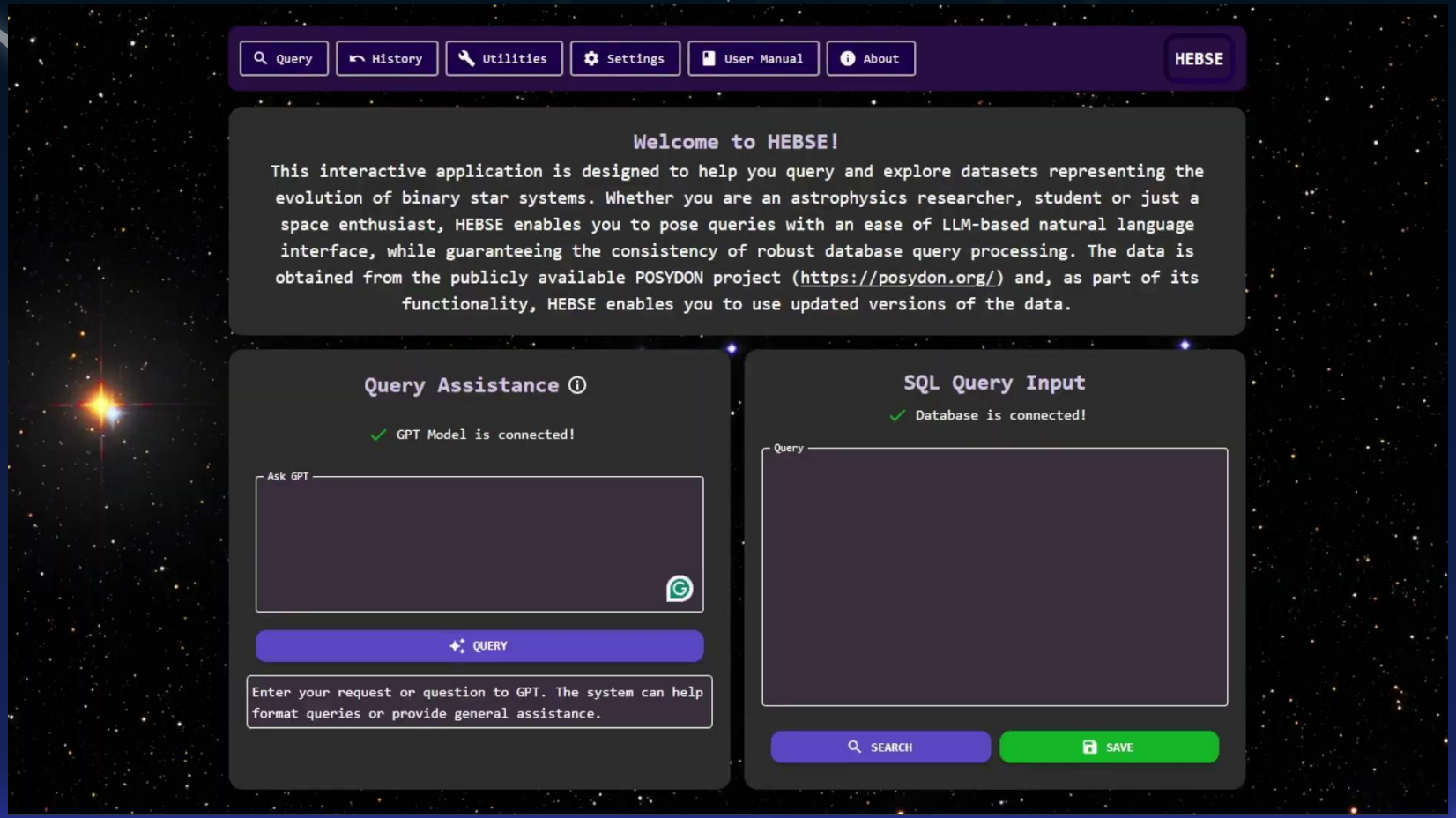
Complex Query Example:
Find all systems which have a mass ratio between 0.5 and 0.7 and at some point an orbital period between 5 and 100 days.

```
SELECT
"binary_history"."model_number",
"binary_history"."age" FROM
"binary_history"
WHERE
("binary_history"."star_1_mass”/
"binary_history"."star_2_mass"
BETWEEN 0.5 AND 0.7 OR
"binary_history"."star_2_mass"
/"binary_history"."star_1_mass"
BETWEEN 0.5 AND 0.7)
AND
"binary_history"."period_days"
BETWEEN 5 AND 100
```

# Project Overview

- Key Features:
  - Import multivariable time-series simulation data into relational database
  - Retrieval of previous queries
  - Enable custom SQL and natural language queries
- Deliverables:
  - Web Based Application
  - Relational Database
  - Data Ingestion Capability

# UI Prototype


Query
History
Utilities
Settings
User Manual
About
HEBSE
Welcome to HEBSE!
This interactive application is designed to help you query and explore datasets representing the evolution of binary star systems. Whether you are an astrophysics researcher, student or just a space enthusiast, HEBSE enables you to pose queries with an ease of LLM-based natural language interface, while guaranteeing the consistency of robust database query processing. The data is obtained from the publicly available POSYDON project (https://posydon.org/) and, as part of its functionality, HEBSE enables you to use updated versions of the data.
Query Assistance
GPT Model is connected!
Ask GPT
QUERY
Enter your request or question to GPT. The system can help format queries or provide general assistance.
SQL Query Input
Database is connected!
Query
SEARCH
SAVE

# Requirements

| Functional | Non-Functional |
| --- | --- |
| • Correctly convert CSV files into PostgreSQL database<br>• Generate functional SQL queries from natural language<br>• Previous and built-in requests are easily retrievable | • Visually appealing and easily navigable user interface<br>• Time-efficient data parsing<br>• Clear presentation of data<br>• Secure data transmission and storage |

# Resource Requirements

- Sufficient storage to maintain a large database
  - Database can be local or remotely hosted
- Sufficiently powerful computer to run the tool
  - More powerful -> faster response times
- OpenAI API key for NLP query assistance
  - Optional, user choice

# Risks Encountered & Mitigations

| Risks | Mitigation |
|---|---|
| Security | Encrypted credential storage, SSH and HTTPS protocols. |
| Limited Dataset Availability | Parsing script designed to strictly follow all known conventions |
| NLP Query Inaccuracies | Thorough testing and training.<br>Full disclosure of generated query to user for vetting with results. |

# Design Decisions

Database management system - PostgreSQL

- Custom data types for data grouping and ease of access

User Interface - React

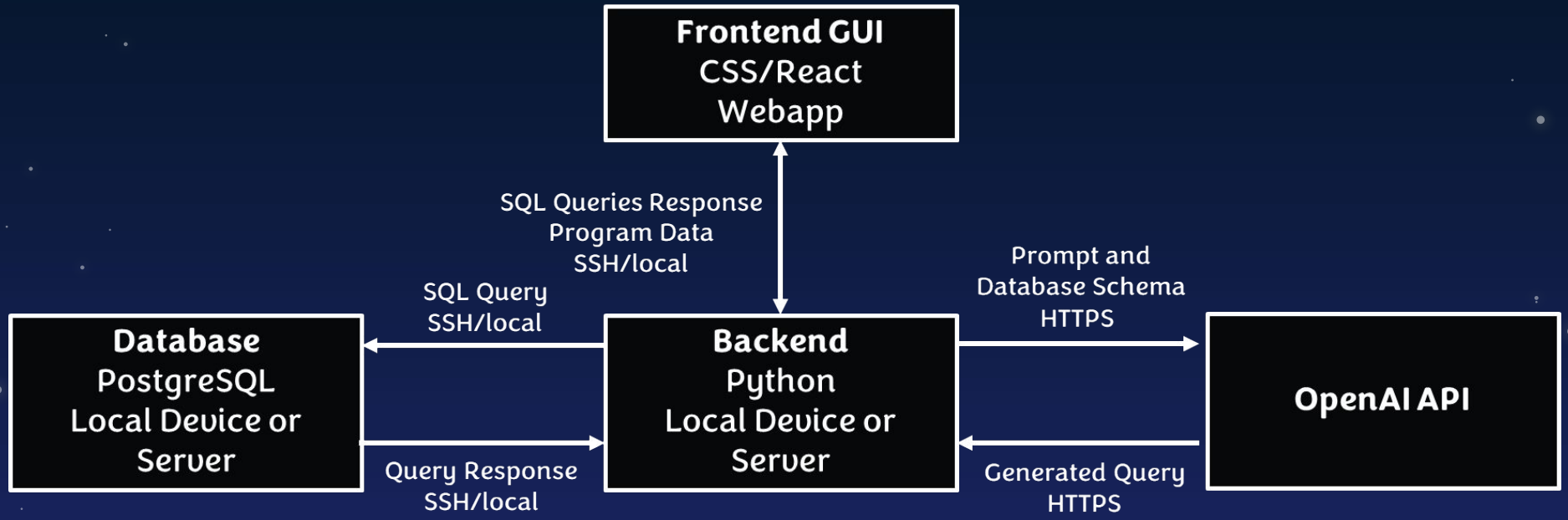
- Modular approach, quick and easy modification of application components

Natural Language Processing - OpenAI API

- Easy integration with applications
- Offloads computational load to OpenAI servers

| System Component | Chosen Technology | Role | Operations | Considered Technologies & Alternatives | Pros | Cons |
|---|---|---|---|---|---|---|
| **User Interface (UI)** | React & TypeScript | Provides user interaction point; captures inputs and displays results | Captures user input for queries, provides dynamic data display, user settings | Angular, Vue.js | Flexible, widely adopted for UI; strong community support | Complex setup compared to simpler UI libraries |
| **Backend** | Python | Central processing layer for data routing, validation, and AI integration | Manages commands, connects components, validates, and processes queries | Node.js, Django | Efficient for scripting and data handling; extensive libraries | May need additional libraries for certain backend tasks |
| **Database** | PostgreSQL | Stores parsed data and allows SQL querying | Supports relational queries and data retrieval from parsed CSVs | MySQL, SQLite, DynamoDB, MongoDB, Neo4j | Robust for handling large datasets; advanced SQL support | Requires setup and maintenance; not as lightweight as SQLite |
| **Data Parser** | Python (Custom-built) | Parses and normalizes raw CSV data and prepares it for database entry | Reads, cleans, normalizes CSV files, estimates missing values, ensures consistency | Pandas, SQLAlchemy | Customizable; optimized for POSYDONS complex data requirements | Requires custom coding for specific data parsing needs |

# Implementation



# Tech Stack

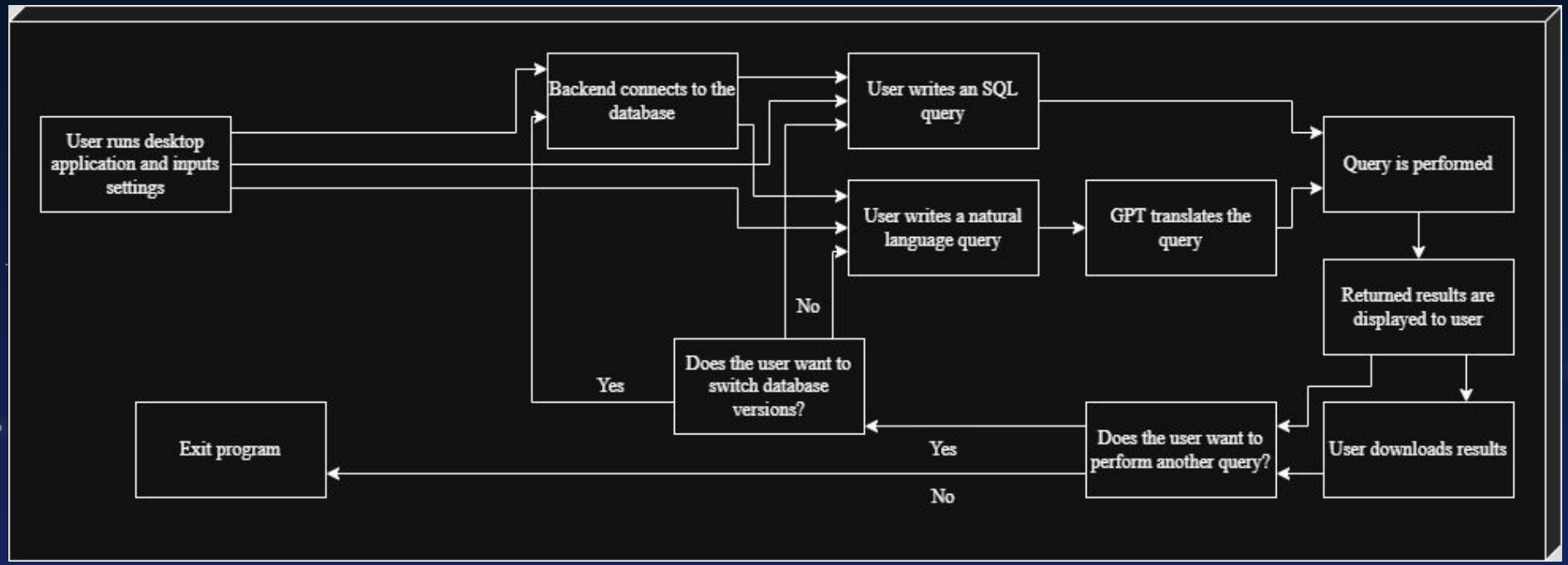
Back End - Python

Front End - React + TypeScript

Database - PostgreSQL

LLM - Open AI

# User Interaction Pathway



# Comprehensive Testing

Unit Testing:
- Vitest (frontend) and Tox (backend)
- 100% coverage

Regression Testing:
- GitLab pipeline utilized to run regression tests prior to merging
- PyTest

Acceptance Testing:
- Client input into all stages of development
- Feedback implementation

# Final Thoughts

**Achievements**

- Finalized an application that exceeds initial goals
- Robust UI for POSYDON integration and offline query support
- Supports local, remote, and hybrid backend modes
- Secured data in storage and transmission with encryption

**Lessons Learned**

- Clear design goals from Fall helped streamline Spring development
- Modular design allows for easier component development
- A visually pleasing and organized UI is key for user experience

HEBSE

# Thank You

## Questions?