Contextualization /Check In

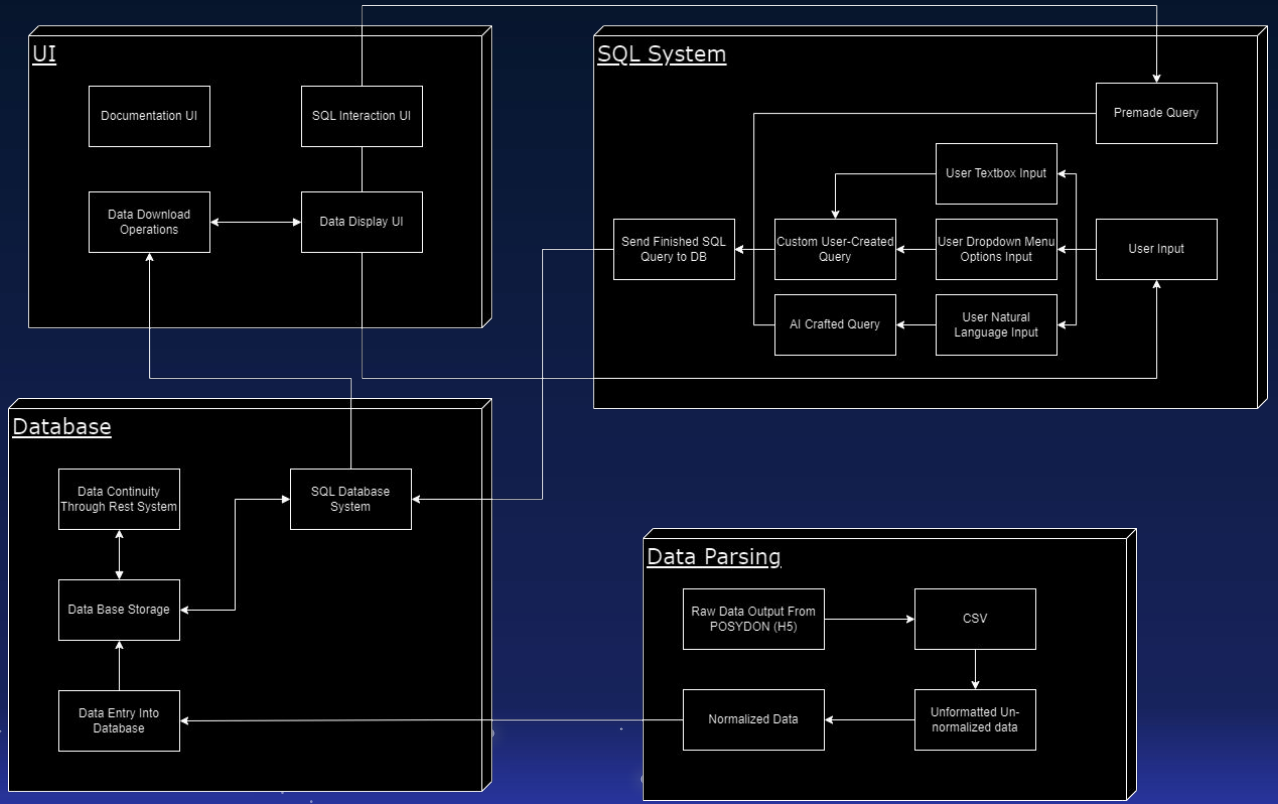
By: Andrew Snyder, Alex Polston, Alek Norris,
Eamon Collins, James Byrd, Svyatoslav Varnitskyy

sdmay25-20

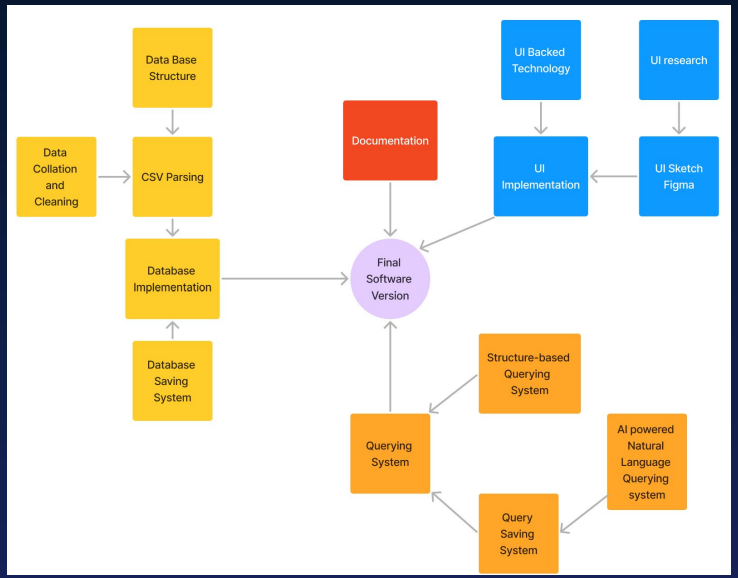
Project Overview

- **Objective:** Develop a system to manage and analyze simulated binary star data
- **Key Features:**
  - Import multivariate time-series simulation data into relational database
  - Provide sample SQL queries
  - Enable custom queries through natural language processing
- **Deliverables:**
  - Relational database
  - User Interface for writing and viewing SQL queries
  - Sample SQL queries

Artifacts - Detailed Design



Artifacts - Task Decomposition



Artifacts - Gantt Chart

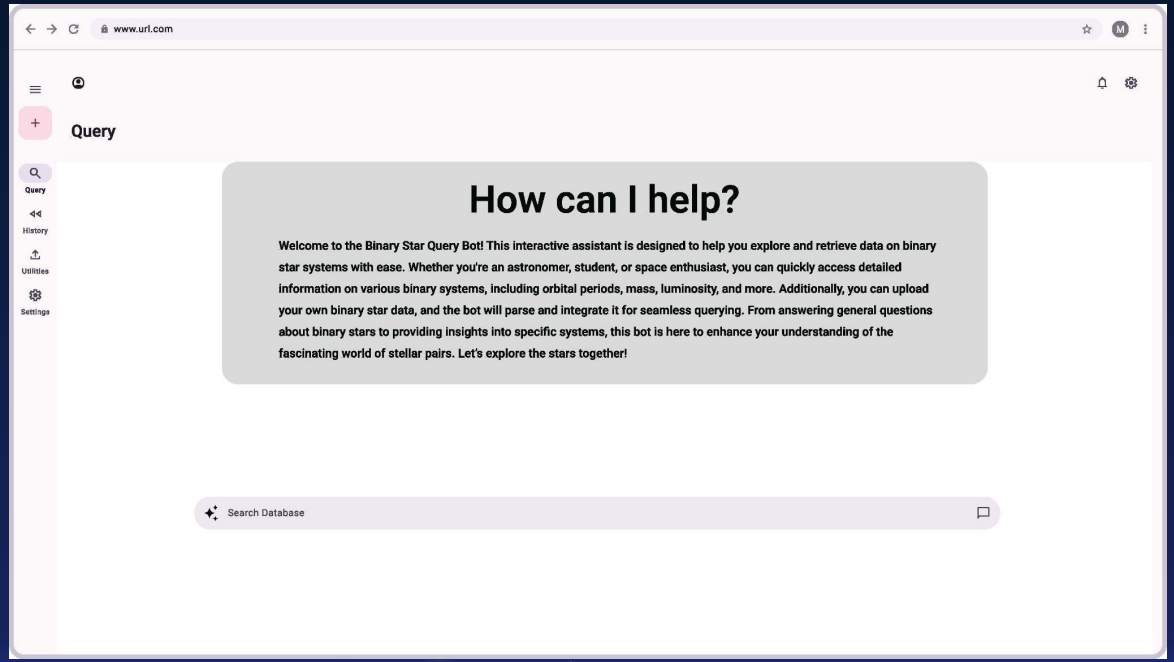
Task Category	Task	10/17 - 24 1	10/24 - 31 2	10/31 - 11/7 3	11/7 - 14 4	11/14 - 21 5	11/21 - 28 6	11/28 - 12/5 7	12/5 - 12 8	12/12 - 19 9
Research	UI Tools									
Research	NLP Tools									
Design Elements	Figma Implementation									
Develop Software	Software Implementation									
Documentation	Documentation									

Task Category	Task	1/20 - 27 1	1/27 - 2/3 2	2/3 - 10 3	2/10 - 17 4	2/17 - 24 5	2/24 - 3/3 6	3/3 - 10 7	3/10 - 17 8	3/17 - 24 9	3/24 - 31 10	3/31 - 4/7 11	4/7- 14 12	4/14 - 21 13	4/21 - 28 14	4/28 - 5/5 15
Research	SQL + Database															
Design Elements	Database Implementation															
Develop Software	Software Implementation															
Develop Software	Database Implementation															
Develop Software	Connection of Database & UI															
Debug	UI and NLP Debug															
Debug	Database Debug															
Documentation	Documentation															

Artifacts - Database Ideation

	Free	Large community for support			Documentation not as extensive	Only available through AWS	Not available for on premise deployment	
	MySQL	High performance, scalability, and flexibility	More robust than MySQL	PostgreSQL	Steeper learning curve	Minimal query options	DynamoDB	
	High loads might hinder performance	Platform independent	Better for higher workloads	More features over competition	Large language support	Scalability	Cost effective	Difficult to join tables
	Graph data model is more natural for queries than relational database	Good performance with large datasets	MySQL	PostgreSQL	DynamoDB	Supports complex queries	Diminishing performance at large scale	
	Neo4j	Scalability issue when introducing new data	Neo4j	Database	MongoDB	Supports multiple platforms	MongoDB	
	Easy to learn	Works well with complex joins				High memory usage	Document size is limited	

Artifacts - Wireframe



Human

Our solution addresses user needs by providing:

- Intuitive user interface
- Accessibility to anyone by allowing queries in English
- Custom queries
- Premade queries
- Query history
- Custom database availability
- Data download

Future considerations to improve solution:

- Host app and data on cloud provider
- Query suggestions

Economic

- Improves upon existing solutions by being more cost-effective and accessible
- Only requires an initial hardware purchase (e.g., processor and hard drive) with no ongoing costs
- Runs on most modern computers, avoiding the need for expensive, specialized hardware
- Potential drawback: additional storage may be needed for the database
- Can be mitigated by using affordable storage options or exploring cloud-based alternatives
- Overall, benefits of lower costs and accessibility outweigh the minor inconvenience of storage

Technical

Internal

- **Technology Integration:** Combines language models, SQL, python, and React for high performance.
- **Flexible Implementation:** Adaptable tools enable custom, efficient design.

External Complexity

- **User-Centric:** React UI with NLP driven queries for accessibility.
- **Market Gap:** Fills a niche need with a unique data management solution.

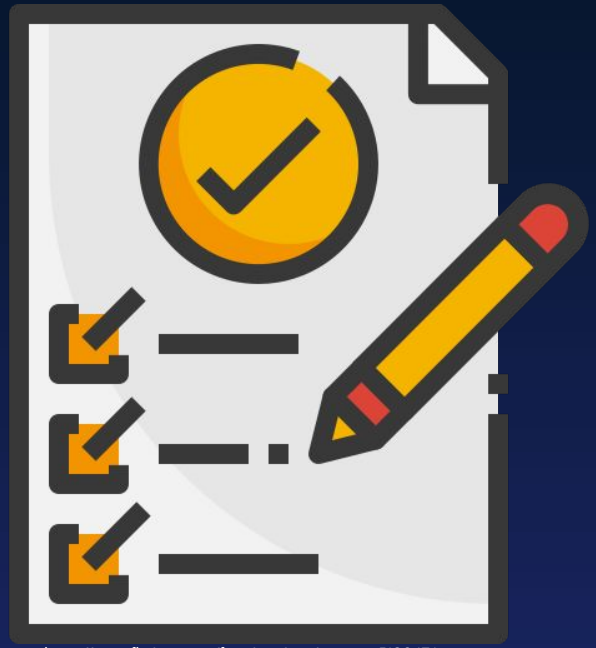
Expertise Showcased

- **Full-Stack Proficiency:** Strong SQL, React, Python, and AI integration.
- **Innovative AI Use:** NLP enhances user experience and accessibility, allowing queries to be made and viable to a new group of users, that poses no prior SQL Knowledge.

Functional Requirements

Functional:

- Supports custom and built-in queries
- Convert natural language into SQL queries
- Ability to save custom SQL queries
- Retrieves data from a database
- Tool to convert compressed csv files into database

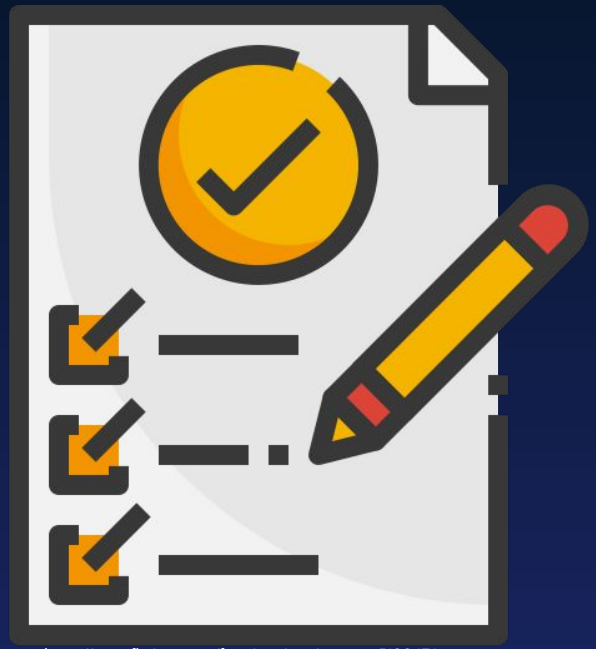


https://www.flaticon.com/free-icon/requirement_5109476

Non-Functional Requirements

Non-Functional:

- Easily understood user interface
- Time-efficient data parsing
- Clear presentation of data
- Queryable database

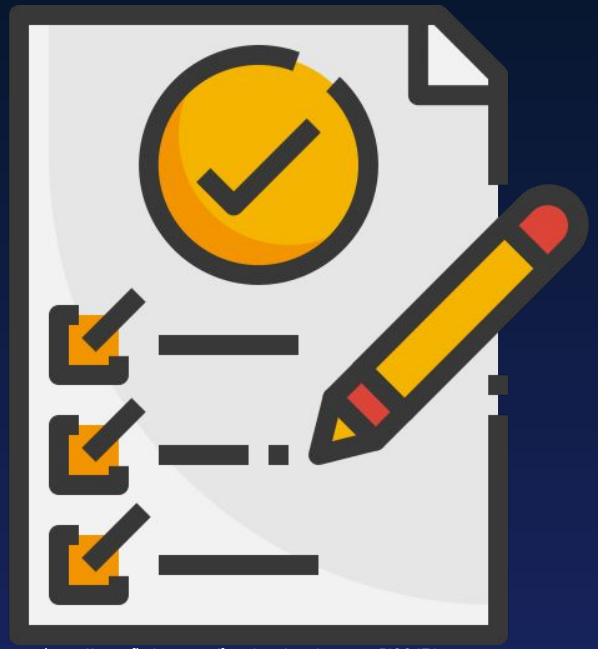


https://www.flaticon.com/free-icon/requirement_5109476

Other Requirements

Economic:

- User needs a computer with necessary system specifications to run the UI
- Computer must have adequate storage for database



https://www.flaticon.com/free-icon/requirement_5109476