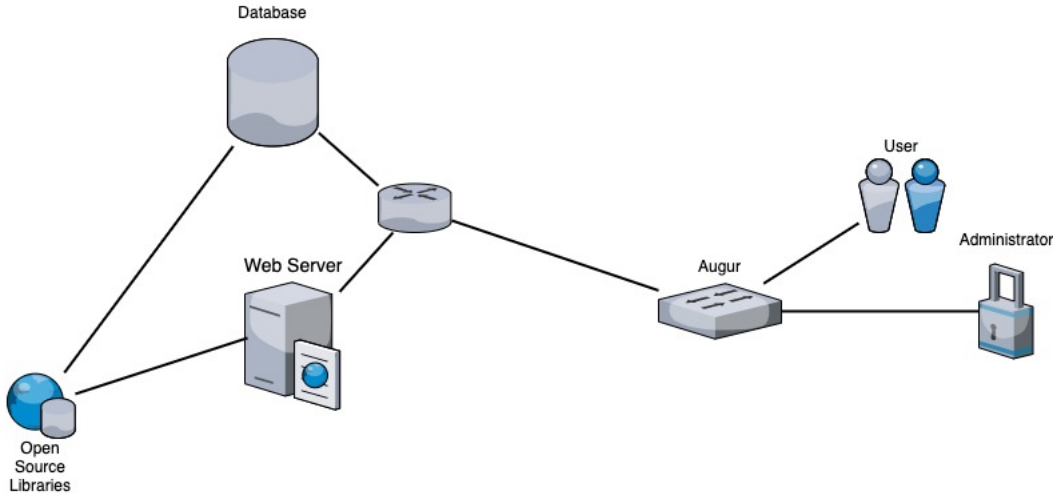
ASSIGNMENT 9

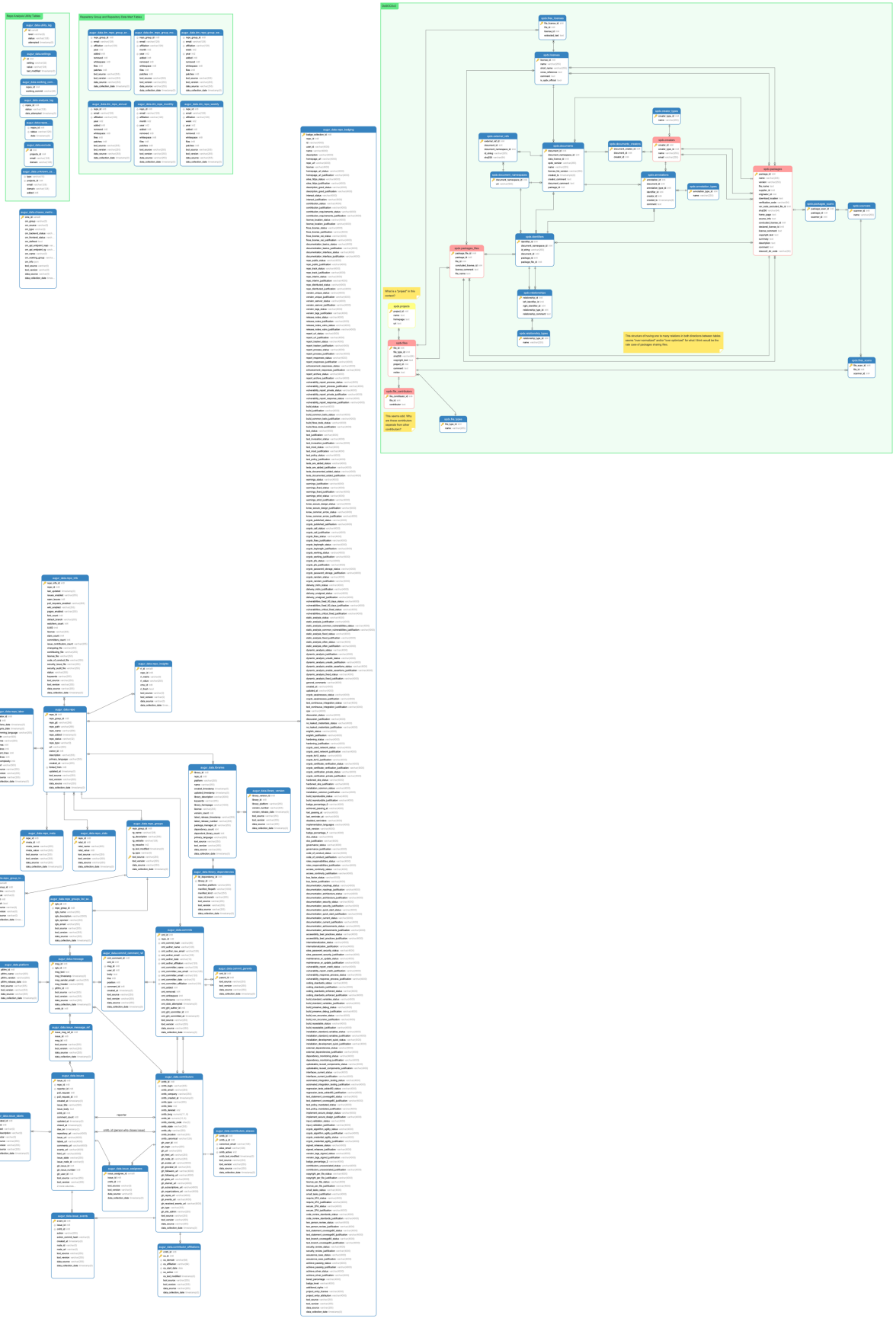
Robert Truesdale RLTGPK

APRIL 1, 2020

- ## Introduction
    - **Purpose**
        - Augur is a project dedicated to aggregating open source data into intuitive charts and models that are able to be understood and monitored by the user.
    - **Scope**
        - Augur is created for those that are in need of monitoring certain repos or repo groups, comparing repos or repo groups, and the ability to download all the data at ease.
    - **Assumptions and Dependencies**
        - Augur is run using a Unix system or Ubuntu (18.04) virtual machine. Augur utilizes PostgreSQL (10 or higher) and MySQL database management systems, which requires a GitHub access token to collect data, and enables read/write permissions. Python 3.6 is required in order to run the API. Users must have Vue-cli, Vue.js, npm, and Node installed to run the front-end locally.
- ## Software Product Overview
    - **System Scope**
        - Augur is open-source, non-hardware specific, and welcomes new contributors. Augur implements REST API structured style, using a slew of technologies listed in the introduction as well as Apidocsjs, Docker, Flask, Gunicorn, Pandas, Sphinx, etc.
    - **System Architecture**
        - External view:



- Internal view:



- **Feature Overview**
  - Augur uses a number of endpoints
    - Batch
    - Evolution
    - Experimental
    - Risk
    - Utility
    - Value
  - Augur enables users to contribute their own metrics and utilize their own custom endpoints.

- **System Use**
  - Augur provides many ways to use the system such as adding repos and repo groups through the backend and utilizing metrics
  - Actor Survey:
    - Head Developer of a large company that manages multiple projects
      - Augur enables this actor to oversee his repo group as a whole, individual repos, and check on the overall health of certain projects.

- **System Requirements**
  - Use Cases:

Use Case Name	UC1 Manager View
System or Subsystem	Augur
Actors	Project Manager
Brief Description	Use case details how to log in and search for repos/ repo groups
Basic Flow of Events	Manager logs in Manager enters desired repo / repo group into search bar located upper left of application Manager presses 'Enter' key to begin search Manager selects desired repo / repo group from the search results

▪ *Use Case Name*	▪ UC2 Insight Review
▪ *System or Subsystem*	▪ Augur
▪ *Actors*	▪ Project Manager / Administrator
▪ *Brief Description*	▪ Use case that details how to view your insights on your repo group, and view individual repos
▪ *Basic Flow of Events*	▪ User log in ▪ User navigates to 'Insights' tab via menu located on the left of the application ▪ User views Most Anomalous Insights Across their Repos / repo groups ▪ User can then click on the information shown to view more insights on their repos / repo groups

    - **System Functional Specification**
        - User can view Repos via repo tab and view information such as URL, issues, commits, names, etc.
    - **Non-Functional Requirements**
        - All users must read the documentation to become familiar with Augur
        - All users are expected to have basic programming language knowledge
        - Users are expected to be familiar with the site due to it's common frontend design

- **Design Constraints**
    - Software:
        - Augur –
            - ran with PostreSQL 10 +
            - requires a GitHub Access Token to collect data
            - runs as a Web App through Vue.js
    - Hardware:
        - Augur is run on Linux or Ubuntu 18.04 or higher.
    - API:
        - Augur has many different APIs used that follows certain guidelines in order to work correctly.
    - Developmental:
        - Augur follows the typical workflow GitHub users follow of branching, committing, and pulling.
    - Download:
        - Augur's files are downloaded as .svg files
- **Purchased Components**
    - Augur has to be run via Linux or Ubuntu, meaning some users may have to purchase a server that meets this requirement.
- **Interfaces**

- **User Interface**
  - Users upon access to the application should see a page created by Vue and Vega-lite and be able to see the data that's been gathered on the hosted instance.