Assignment 9: Individual Requirements Analysis: Spring 2020 CS4320/7320 Software Engineering

Alex Webber

Introduction

The open source software ecosystem consists of hundreds of thousands of different projects. It is often difficult to determine if an open source project is progressing well, and if it is worth investing time and/or money into. Thus, there is a demand for a way to understand the sustainability of an open source project. In order to do this, we must have a wide array of reliable metrics of open source software. The CHAOSS community develops these metrics, as well as visualizations and software to implement them, in order to make it easier to see how a project is progressing. The purpose of this document is to identify the software requirements and structure for Augur, an open source software by CHAOSS for health and sustainability metrics.

Software Product Overview

Augur a web application, Python library and server that provides different metrics on open source software pertaining to project health and sustainability. It makes it easy for companies to obtain relevant data on open source projects, as well as visualizations to make said data meaningful. Augur has its own metrics already defined, but it also gives you the ability to create your own metrics. Data can be obtained through Python commands or directly from the augur REST server. The web application also gives pre-built visualizations of metrics on repositories and repository groups.

System Use

User's of the Augur system will be generally split between two groups: developers and end users.

Actor Survey:

- **Developers:** Developers in this context are any people who will use Augur for data comparisons, creating metrics, and extracting data. They can compare data from certain metrics between different repositories. They can

create their own metrics, and they can use these metrics to get data on any/all repositories. Developers can, and often will also use this data to create their own visualizations, although Augur does offer it's own visualizations

- **End Users (visualizations):** The end user's of Augur in this context will be people who are not actually writing code or making GET requests for data. They will typically only interact with the system to download visualizations or sets of data. End users will interact with Augur through the web application.

System Requirements

System Use Cases:

Use-case name	**Developer creating metric**
System	Augur
Actors	Developer
Brief Description	This use case explains how a developer creates their own metric, by first creating a metric function to obtain the data, and a metric endpoint to store the data.
Basic flow of events	Basic flow begins after the user has already installed all of augur. Within the augur database, they can begin making their own metric. 1. User writes a PostgreSQL query against Augur's schema to extract their desired data 2. Wrap & parameterize the SQL query in a metric function in the **augur.metrics** module 3. User implements metric endpoint by using the relevant addMetric function (e.g. addRepoGroupMetric) and the metric function they created in step 2. 4. The system will then store the metric data in

	the API endpoint they defined.
Special Requirements	Usability: The system requires user's to have functional knowledge of PostgreSQL and Python.
Pre-conditions	Augur and PostgreSQL must be installed
Post-conditions	The created metric data is stored in the Augur database and is accessible through the created API endpoint

Use-case name	**End User downloading visualization**
System	Augur
Actors	End User
Brief Description	This use case explains an end user (such as an organization) can download a visualization on a repo made by augur
Basic flow of events	Basic flow begins by the user providing a Github URL of a repo they are interested in. 1. User provides Github URL 2. Augur runs metric functions on the repo 3. Augur stores the metric data 4. Augur creates a visualization of the data 5. User is presented with the visualization to download
Special Requirements	User must have a URL to a repo or repo-group
Pre-conditions	None
Post-conditions	The visualization is downloadable as a .svg file

System Functional Specification:

Augur's web application will implement the system architecture with a client-server approach. Augur will have separate functions that are accessible through running Python commands within Augur, instead of using the web application. These commands will facilitate the client to retrieve data from the database, and also to add data to the database. Augur supports most UNIX systems for this.

Non-Functional Requirements:

1. The web application will be easy to navigate and use for anyone, no prior knowledge or experience should be expected.
2. Augur will have installation instructions that are easy to follow for anyone who has experience with Python.
3. Data and visualizations should be easily downloadable in common formats, e.g. .csv, .json, etc.

Design Constraints

1. The web application should run on all major browsers - Google Chrome, Firefox, Microsoft Edge, etc.
2. Augur requires Python 3.6 or higher
3. Augur requires PostgreSQL 10 or higher
4. Augur requires Vue.js, vue-cli, node and npm
5. Good coding practice such as cohesion and loose-coupling should be used at all times

Purchased Components

All software used for this product is open-source, no licenses are needed.
Hardware:

1) Web server
2) Database server

Interfaces

User Interface:

User interface is available without authentication at the website. AJAX should be used to asynchronously send data from the server to the web application. The design of the user interface should be straightforward and easy to follow.

Software Interface:

The web application will communicate with the REST server and Python library.