Jessie Murphey

Individual Requirements Analysis

Assignment 9

Overview

1. Introduction

1.1 Purpose

When dealing with thousands of open source projects it can be extremely difficult for companies to see which projects are still being maintained as well as which projects are likely to be abandoned. The purpose of this document is to specify the requirements and software needed to allow technology companies and IT organizations to make sustainability predictions toward current projects, as well as evaluate the amount of previous and current changes to a project and by whom.

1.2 Scope

The scope of this document is to describe the proposed system, as well as any system requirements and design constraints.

1.3 Assumptions and Dependencies

The platform would be dependent on a source control for the open source projects to be tracked using Github/Gitlab, as well as a web server that is most likely cloud hosted.

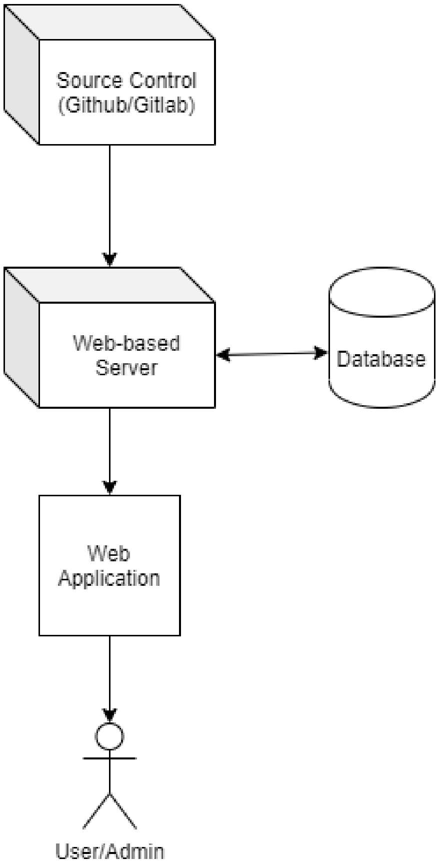
2. Software Product Overview

2.1 System Scope

The proposed system would allow companies to select the open source project they wished to evaluate. They then would be shown a graphic of the number of commits since the creation to present or based on the selected year/month/week. To the side the software would present the user with a list of top contributors, with the number of commits, and the total number of contributors that have worked on the project. This will allow companies to evaluate if a project is currently still being maintained as well as the “Bus Factor” allowing them to more accurately make a prediction on the projects’ sustainability.

2.2 External View of the Product

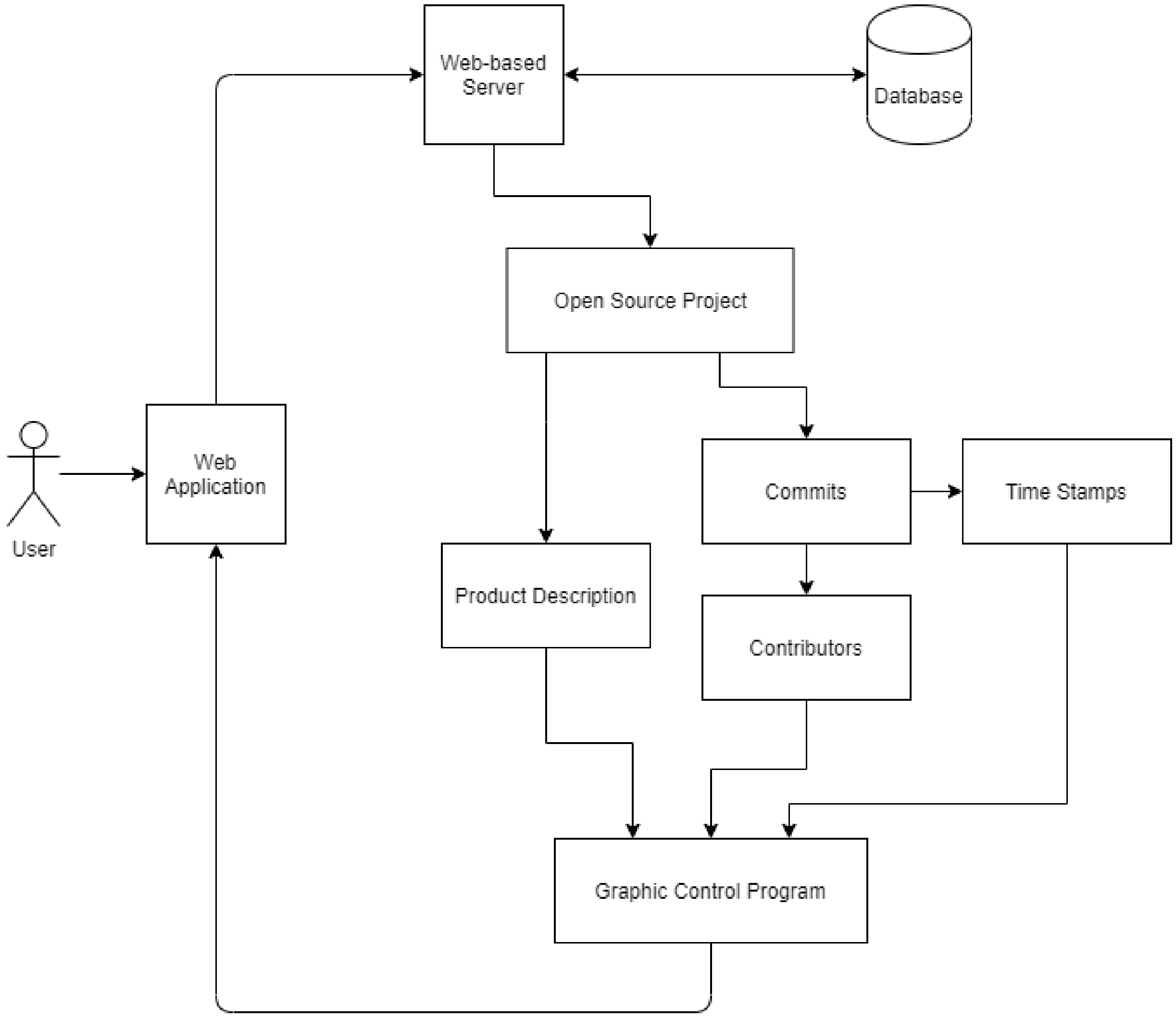
The diagram below shows the external view of the product. It is implied that an internet connection is needed for the source control, web-based server, database, and web application to communicate.



2.1 Internal View of the Product, Broad Use Case Example

The diagram is below is a broad Use Case that shows the internal view of the product after the user has requested a specific open source project to view through the Web Application. It is implied that the source control has been updating the database beforehand. The Database will pull specific information from the specified Open Source Project which then will be processed through the graphic control program and presented back to the user through the web application.

**It should be noted that this program is just to view and evaluate open-source projects, not to modify them.



3. System Use

3.1 System Use Description

The system is intended to only be used to create a graphic for the user to evaluate the open source project. All admin services, such as adding, editing, and deleting repositories (open source projects) are not handled by this system.

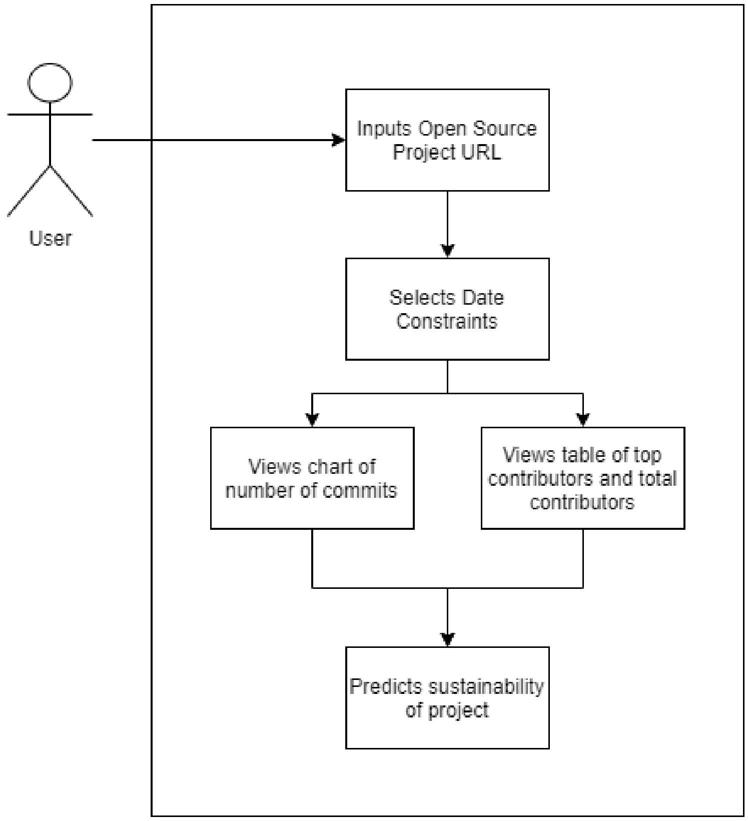
3.2 Actor Survey

User:

- Select specific open source project
- Select date for data constraints (If they only wish to view a specific time period, year/month/week)
- View the open source repository statistics

4. System Requirements

4.1 Specific User Use Case Example


User
Inputs Open Source Project URL
Selects Date Constraints
Views chart of number of commits
Views table of top contributors and total contributors
Predicts sustainability of project

4.2 Feature Specification Overview

Users:

F1: Search for a specific open source project by URL

F2: Present User with a graphic (line graph most likely) of the number of commits by date.

F3: Default graph will be the overall commits since repository creation.

F3: The system can allow the user to constrain the graph to specific dates, such the year, or a specific month or week.

F4: The graph intervals will dynamically change according to the date constraint

or by how long the project has been active. (Change the scale to years, months, weeks, days)

F5: To the right of the graph the system will present a table of the top contributors' IDs or names and the number of commits they have added.

F6: Underneath the table will be a running total of the number of contributors and the total number of commits by everyone.

F7: All the system data, as far as number of commits and contributors, will dynamically change as the repository is updated by the users. The editing will happen outside the system.

4.3 Non-Functional Requirements

- The user should be presented with a search engine to place the repository URL
- The graph and table should be easy to read, and the data should be presented clearly
- The date selection should only go up to the current date
- All UI elements should be intuitive and easy to use and understand

5. Design Constraints

1. The software can only pull data from already created repositories
2. The repositories the software pulls from need to agree to be tracked by the source control
3. All changes to the repositories should happen outside the program
4. If a repository is deleted the system data on it should be deleted as well (although if the creator wishes we might be able to implement it so it is archived instead).
5. The date selection constraint for the data should only be available from the time the repository was created to the current date.
6. The software must update the web application after any changes are made to the repository that affects the relevant data.
7. If it is needed the system might have to implement a login system that allows the user to only see repositories, they have access to. This will add security to analysis data, however since these are open source projects, I'm not sure if this would be necessary.

6. Purchased Components

- Virtual servers on a cloud platform
- (Optional) Additional separate database for archived projects

7. Interfaces

The data is mainly handed by the graphic controller which is then presented to the web application on the web-based server

- Dynamic reporting of repositories for contributors, time-stamps, and commits
- The website application that will present the data in a way the user can understand