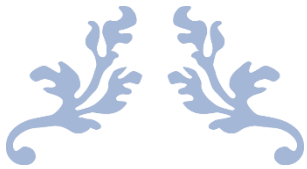


ZHENG LIANG

Individual Requirements Analysis

Project: Chaoss

Introduction:

Background: The importance of open source is self-evident, and everyone's default is to recognize the importance of open source to the world we live in. The only question is how to profit from the fact that the economy of ownership limits the way many people think. Do we need to accept the moral responsibility? What's more, as the open source ecosystem takes shape, there are more problems and people pay more and more attention to it. For example:

Contributors to open source want to know which open source projects have impact, and where they should go? The open source Community wants to attract more members, ensure consistent quality, and reward those who make significant contributions. To solve these problems, CHAOSS was born.

Purpose:

1. Establish implementation-independent metrics to measure Community activity, contribution, and health
2. Develop and integrate the corresponding open source software to analyze the development of the Community
3. Build replicable project health reports

Open source development index, methodology, and the corresponding software project, aiming to make open source projects can be benign and sustainable development, through measure the health of the open source project and sustainability. Further, CHAOSS will seek to improve the open source project the maneuverability of the health and sustainability, and let all the stakeholders can make a more informed decision

Scope: Chaoss is free to download and use. Anyone who use this project might Potential contributions to the open source program.

Software product overview

Augur allows users to look at repo groups they are in. They can also get the information including data for the health and sustainability of the repo on vue. The users can view for github from their repository. This allows open source software teams to compare their projects with other projects using metrics, allowing the team to analyze the project over time and provides insight into how the project compares historically with other projects. It gives users an easy way to download data and analyze the data.

System Use, including an actor survey

Actor survey

Contributor: Contributor are the key to augur. They can add to any project they want, they can fix and improve the projects they join in. Their change which can be updated if the administration confirmed. They can also create a project and waiting for other contributor involvement, which will improve and fix their own project.

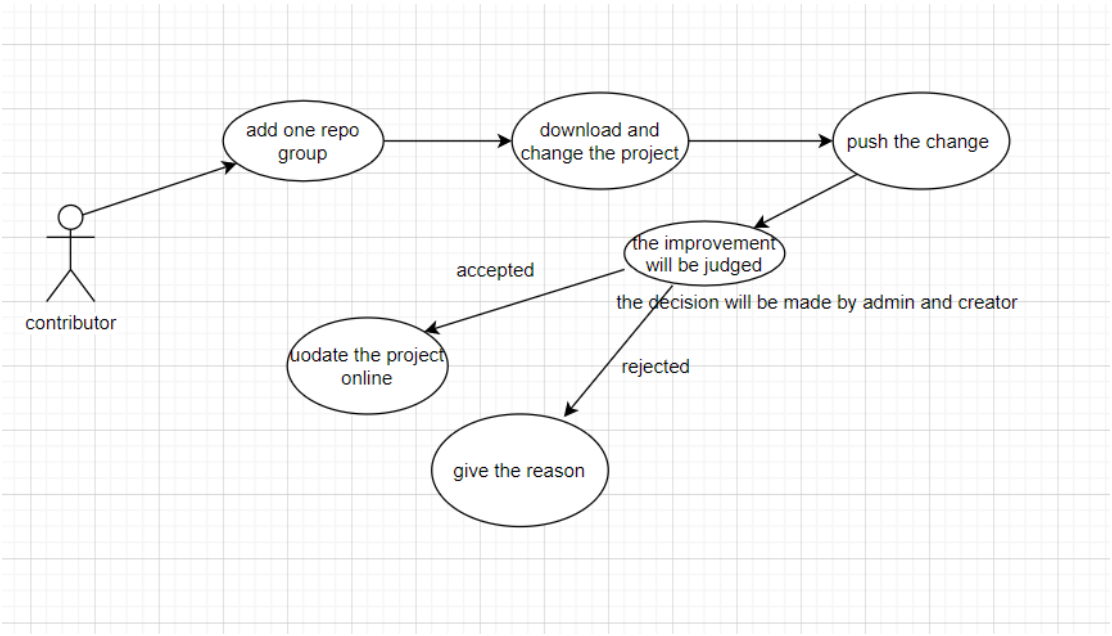
User: Users can only browse or reference projects, but they don't get the permission to directly change the content of the project. They can't change the content unless they transform their identity into a contributor

Admin: administrator will review the changes and negotiating with the creator to incorporate the changes into the original project, it will also protect the interests of the creator and contributors.

System Requirements (including 2 use cases, a system functional specification, and a list of non-functional requirements)

Use case 1:

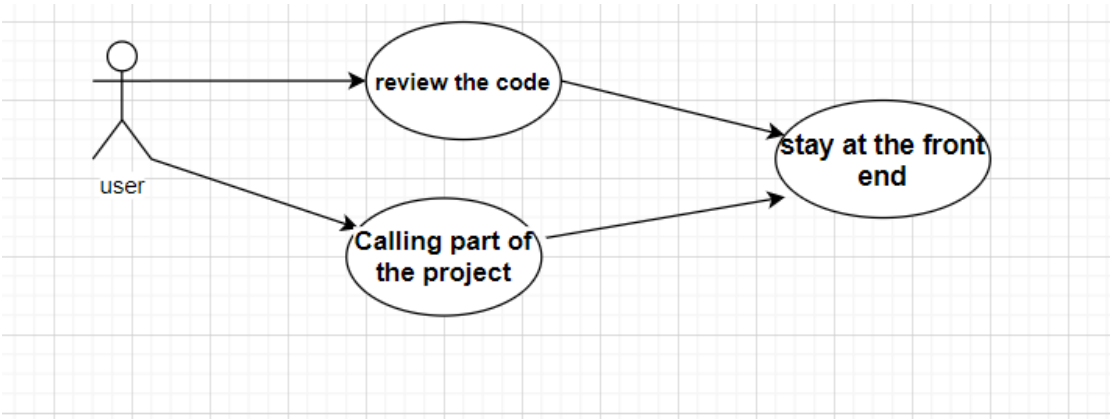
Modifying open source projects



Actors: contributors

Description: the contributor needs to add repo group and download it. the contributor needs to wait for the permission if he wants to fix or improve the project onlilne.

Use case 2: review and refer the project



Actors: user

Description: unlike the contributor, the user doesn't have the privilege to change the content of the project. But users can also review and use the

project by adding the repo group. But it needs to apply for becoming the contributor.

system functional specification:

1. The software will work on web browser and it will be up to date.
2. Users should have different permission and need a login page.
3. The pulled information should match for the github.
4. The system should work on both OS including linux, windows and so on.

Non-function requirements:

1. The website's database should be able to store repo groups data.
2. The server can handle multiple users logging in at the same time
3. Websites need to be protected against malicious intrusions including SQL injection , ARP attack and so on.
4. The website needs database maintenance including Backup system data and restore database system.
5. Data loading should be quick

Design constraints

1. The website should work on all kinds of web browser.
2. All operations should be allowed on the command line including git bash on windows or command in Mac.

3. The database is secure from non-administrative access
4. There should be a data cache when an error exits the page
5. When the page loads, it should be loading massage to make sure the user knows about the process

Purchase component

1. Cloud storage space like amazon cloud, the cost is based on the size of the server and number of users. It's expected to be around $50 per month.
2. Domain name of a website, 100 dollar.
3. maintenance cost
4. equipment cost, including computer and internet, thousands of dollars.

Interface

1. login page. It is necessary for all users to login, there should be no way to skip this page.
2. Home page. Any page can be returned to home page directly
3. Admin interface, the admin page may have more function than normal user page.