Software Design Document

2/7/20

Version 1.0

Project Sponsor:

Dr. Igor Steinmacher

Team Mentor:

Fabio Santos

Team Members:

Adriana Aguilera, Joseph Danciu, Jevin Dement, Ugo Dike, Stavros Triantis

Table Of Contents

1 Introduction

In the early days of computing, obtaining the latest software often came with a hefty price tag. Today, consumers are presented with many free alternatives to costly mainstream software releases, all thanks to open source software. The term "open source software" (OSS) refers to software that is free for the public to "inspect, modify, enhance, and distribute [1]." Open source software is cost effective, updates regularly, and hits the market relatively quickly. Some of the most well known open source projects include WordPress, Mozilla Firefox, Audacity, and OpenOffice. A recent poll from opensource.com discovered that "71% of people surveyed" were expected to use open source software day-to-day in their developer jobs [2]. This type of software is often produced collaboratively via a public platform such as GitHub through the hard work of community developers.

Open source software is not only beneficial to consumers, but to developers themselves. OSS offers a sense of community, a chance to improve software development skills, a way to learn new technologies, and adds to resumes. Unfortunately, the road to contributing to projects isn't always straight forward. Often newcomers are overwhelmed by the sheer amount of projects available. This information overload is often discouraging as they do not know where to begin contributing. As mentioned prior, these developers are an essential part of the OSS community. A loss of a potential developer is a loss to the greater digital community as a whole.

Our client, Dr. Igor Steinmacher, is a researcher and assistant professor at Northern Arizona University who has devoted much of his research into the field of open source software. Dr. Steinmacher and his colleagues want us to create a tool that will help reduce information overload at the user end and encourage newcomers to contribute.

To accomplish this task, our team, alongside Dr. Stienmacher, will create a web application with an easy to utilize user interface that will carry out the matching of countless developers to ideal projects based on their skill set. This will solve the tedious process of finding projects that best suit an individual newcommer's skills and lead to more OSS contributions.

There are many user-level requirements for accomplishing this task. First, the application users will login to their GitHub profiles from our website. This will allow us to access their past contributions, so we can later match them to appropriate projects. They will have the opportunity to fill out a questionnaire, which may fill in skills that we are not able to mine through their GitHub profiles. Lastly, they will be able to view their unique matches through a section on the webpage.

At a more functional level, there are many requirements we must implement, allowing for a quality user experience on our webpage. The most important requirements elicit checking if a project is written in Java, cloning repositories, running mining tools, inserting information into a database, matching users to projects, and displaying the matches. All of these mechanisms are being held to a high performance standard. This includes a promise that both user skills available OSS project skill requirements will be mined within two hours. Lastly, matches will be saved to each user's profile for access at any time, where the list of matches can be added to. While we have a high performance standard, there are environmental constraints we must account for. This entails a 90 day limit on the mining of user data, and the recommendations only apply to Java based projects.

This document will entail a comprehensive explanation of our envisioned solution to bring our client's product to life. Furthermore, we will include detailed documentation of our

plans for implementation, timeline, architectural design, components, classes, modules, and interfaces. Next, we will break down how we plan to implement our product.

2 Implementation Overview

In the previous section, we defined OSS and discussed the problems many face when attempting to contribute. We also gave a brief overview of the user-level, functional, performance, and environmental requirements. In this next section, we will give an overview of our application by painting a "Big Picture" of what we are creating and the technologies necessary.

Match Source will be creating a web application that will empower developers, and give them the confidence to contribute to OSS projects by providing them with a list of projects that match their skill set based on information collected.

Figure 1

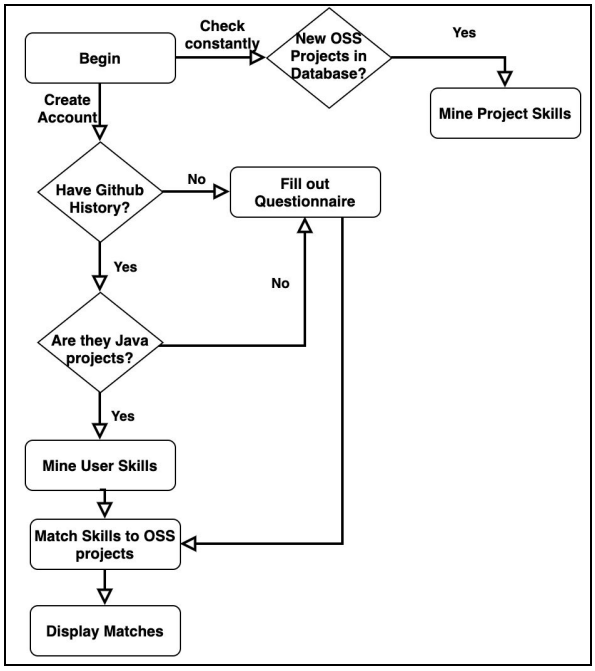

Figure 1: *A Flow diagram showing the logical breakdown*

The way our application will work is very straightforward. The user will be required to sign up to make an account and then have two options to consider. Either they provide their GitHub username or fill out a questionnaire asking them about what skills they contain. Since some developers might be new, or don't have a GitHub account, we will have a questionnaire presented to them that will have them pick the user skill that they have experience in. If the user has a GitHub account they will be able to login and have their account authenticated. Our parser will then gather key information from the GitHub API, which displays information in JSON format, most importantly, the projects that they have contributed to. The information collected will go through a mining process which extracts the imports a user has in the projects that they have contributed to or available in their repository within the last 90 days. After this process, the matching process will be executed by comparing the user skills to the available projects. A list of matches is generated of potential project matches and presented on the webpage.

Figure 2

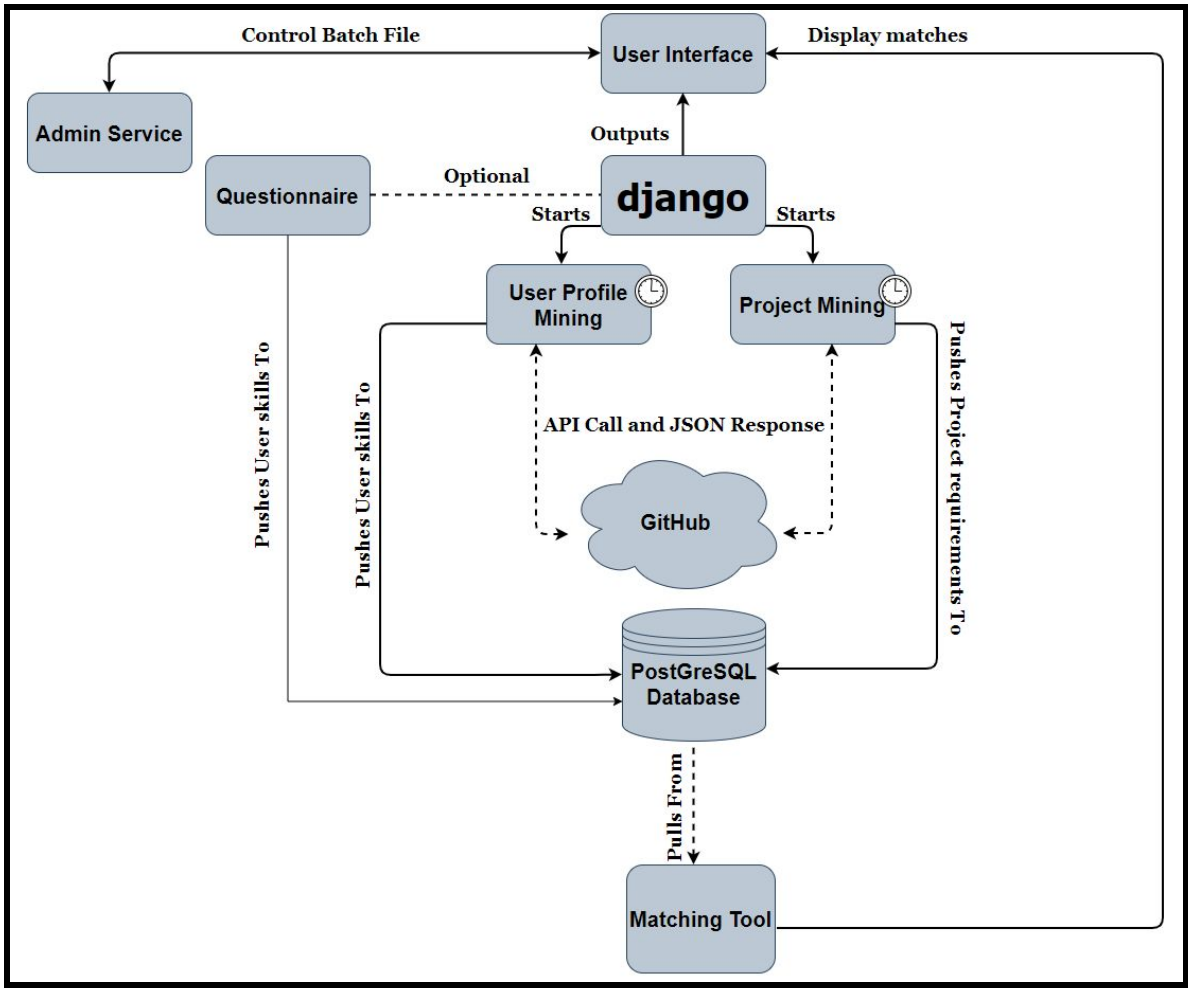

Figure 2: *A diagram showing the tools we will be utilizing for our product*

The main technology that our application will revolve around is the web framework Django shown in Figure 2. Django is a Python model view template web framework that provides code for common operations like database manipulation, HTML templating, URL routing, session management, and security which make the process of creating both great frontend user interfaces as well as fast and reliable backend functionality. We have chosen for our backend logic as well as our frontend user experience to be done using Django due to its compatibility with Python in general and the existing support and libraries that can be utilized

when using Python and Django. Django is a very powerful framework and will allow us to do everything that is needed for this product while doing so with high efficiency and a streamlined development experience.

We will be storing a large amount of data for our application in order to be able to match user skills to available projects. To store the data retrieved from projects and user skills we will be using PostgreSQL due to its ability to handle heavy workloads and its smooth integration with Python and Django. PostgreSQL is an open source relational database that supports both SQL for relational and JSON for non-relational queries.This is important since we will be importing plenty of json files since that is how the data is retrieved from the GitHub API.

The final piece to this application is web hosting. In order to make this application accessible to newcomers of OSS, we will be using a service called Digital Ocean. Digital Ocean offers an affordable, reliable, and customizable service, with a variety of features unique to most web hosting platforms. With such a reliable service, our customer will be able to access our web application at any time.

3 Architectural Overview

In the previous section, we discussed the implementation overview of our application. This section will contain high-level detail about how the system will be constructed, to better understand the components that comprise the final product.

Figure 3

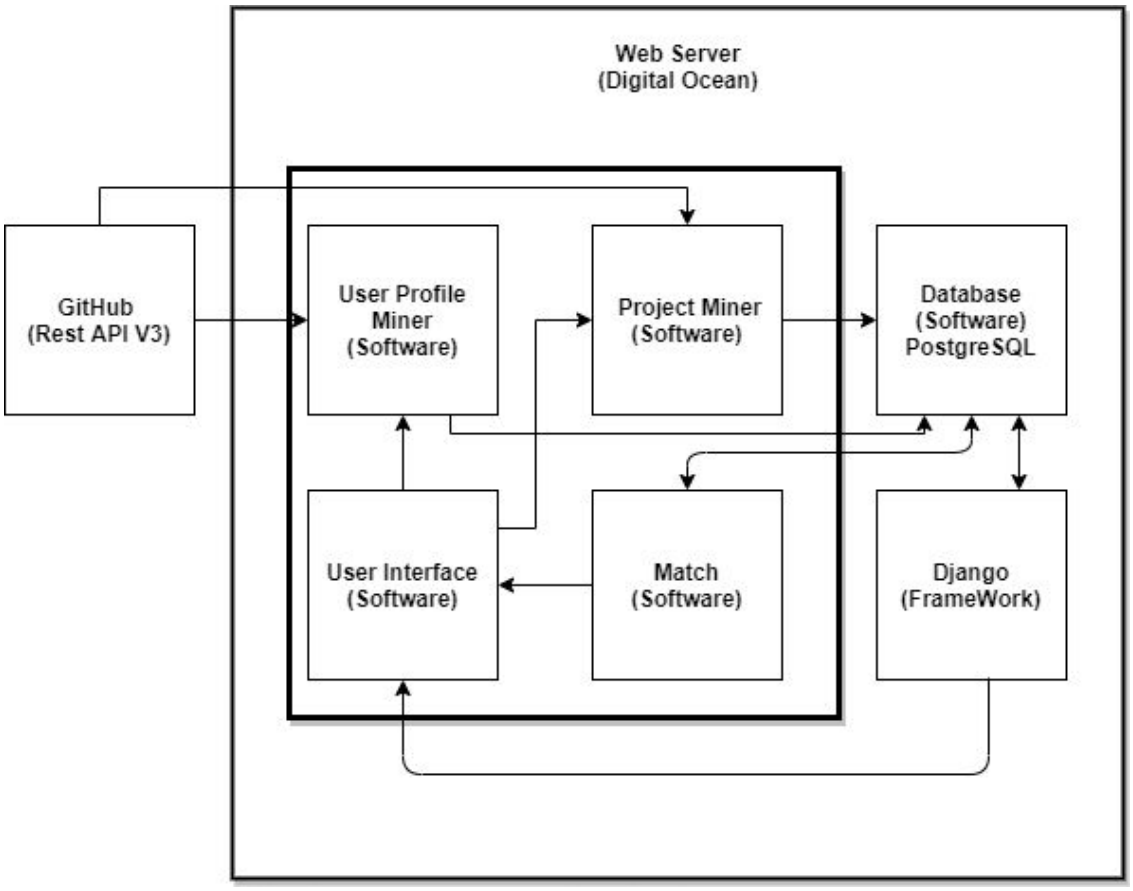

Figure 3: *A diagram portraying the architecture of our product*

Using Figure 3, we will break down the main components that comprise our systems architecture. The diagram contains an inner and outer component, where the inner component is a sub component of the outer.

The outer component is the Web Server we will be using; which includes most of the important components of the architecture we will be building. This includes our PostgreSQL Database, Django framework, and sub-components. The Django framework will allow us to create a user friendly interface that will display the matches. All information used in this application will be stored in a manageable PostgreSQL database.

The inner component is comprised of the User Profile Miner, Project Miner, User Interface, and Matching Tool. The User Profile Miner will be in charge of mining GitHub user pull requests events to extract their skills from projects they have contributed to. The Project Miner will mine through a pre-populated database containing OSS projects available on GitHub to extract skill requirements to contribute. The Matching Tool will pull user skills and available OSS project skill requirements to match. Lastly, the User Interface will allow the users to interact with our application and have matches displayed on their personal profile.

The only aspect of our architectural depiction that is not contained within the outer component is the GitHub Rest API. The User Profile and Project Mining Tools will extract information from Github using Version 3 of GitHub's Rest API.

4 Module and Interface Descriptions

Above is a component diagram of our project. This diagram consists of eight of our most important components. Each component works together in unique ways that will be described below as we break down our modules and interface.

4.1 Web Server

The provider for our web server is DigitalOcean, as our client already is subscribed to this service. The web server will be the vessel for our web application on which it will be hosted, as well as provide a space for our database.

4.2 GitHub API V3

The GitHub API is the means of which we are able to mine data from user profiles and retrieve information about open source projects. Specifically, we will be utilizing version 3 of the API, as it is the most straightforward to use and access when implementing Python. This version provides a JSON file with the specified information requested.

Interaction between User Profile Miner: The user profile miner will use GitHub's API to search through a users profile, after they have provided us with their GitHub username, we look for Java contributions and then proceed to clone their repositories.

Interaction between Project Miner: The project miner will accept GitHub repository URLs to later be added to the database and suggested to other users. The miner will use the API to check if the URL provided is valid and is a Java based project.

4.3 User Profile Miner

The User Profile Miner is a program which will allow us to extract unique skills from each user's GitHub profile. The program will check for the java language based projects and download the repositories of those projects. It will only be mining projects that have been contributed to in the past 60 days, to ensure that each developer still has proficient knowledge in these categories.

Interaction between Database: After these projects have been mined, they will be sent through an outsourced project miner and the skills found within the project will be saved into a PostgreSQL database.

4.4 Project Miner

The Project Miner is a program that will have the ability to manage OSS projects listed in a database. There will be functions to Add projects, and a function for the admin to update projects. Projects added to the manager will be mined using the outsourced project miner. The mined "Skills" from this portion will be considered the required skills to be able to contribute to that specific OSS project.

Interaction between Database: The skills found within the mined projects will be saved in to a PostgreSQL database.

4.5 Match

The user's skills and OSS project skill requirements will be compared. If there is a match between the skills and requirements, we will display it. All matches will be listed for the user, along with other information so they can start contributing.

Interaction between User Interface: Users can initiate a new match request on their profiles. After matches have been finalized, users will be able to find them in a list once logged into their profile. An email notification will be sent to the individual's email once the mining and matching process has completed. Matches will be stored in a profile indefinitely, and users will be able to access them on their own time. Projects will only be removed if a user no longer meets the skills necessary.

Interaction between Database: The matching tool will have the ability to pull data from the database, specifically user skills and project requirements to test for a match. It will also have the ability to push its final matches to the database.

4.6 Database

We will have two separate sections of our database, one for storing user information and one for storing project information. The database we have chosen to use is Postgresql, as it integrates well with our project and supplemental code provided by the client.

Interaction between Match Tool: As mentioned above, the match tool will be able to pull user skills data and project requirements data, as well as push its final matches.

Interaction between Django: We will be able to query skills from the database that will be used to populate the the questionnaire.

4.7 User Interface

We will be using Django, alongside HTML and CSS, to build our frontend as well as connect it to our backend. In the following section we describe the major components that make up the user interface and make the modules functional to the end user.

Interaction between Django: Django will be used to write and create a user interface for our product.

Interaction between User Profile Miner: A form will be used to allow the user to input their GitHub username. The user profile miner will receive that form input and begin its process.

Interaction between Project Miner: Similarly to the profile miner, the project miner will use a form to receive a URL. That URL will be passed to the project miner; from

there it will start its process and either output an error message or a “project successfully downloaded” message.

4.8 Django

Django is a Python based web framework that allows us to build both the front end and back end components of our application.

Interaction between Database: Django is very good at handling database driven web applications, such as our product. We will use Django to integrate our Postgresql database, scripts, and user interface.

4.9 UML Class Diagram

Below is a class diagram that organizes and connects our modules.

Figure 4: UML Class Diagram

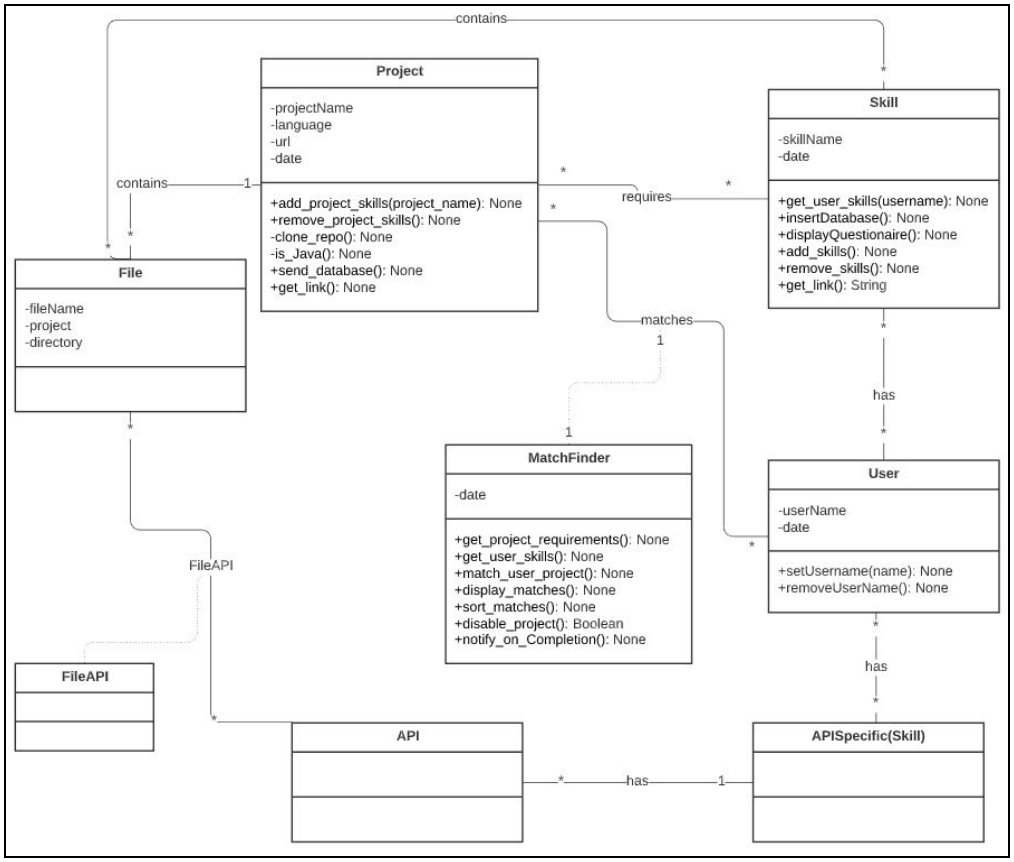

Figure 4: *A UML Class Diagram showing the flow between classes*

5 Implementation Plan

Figure 5: Implementation Chart

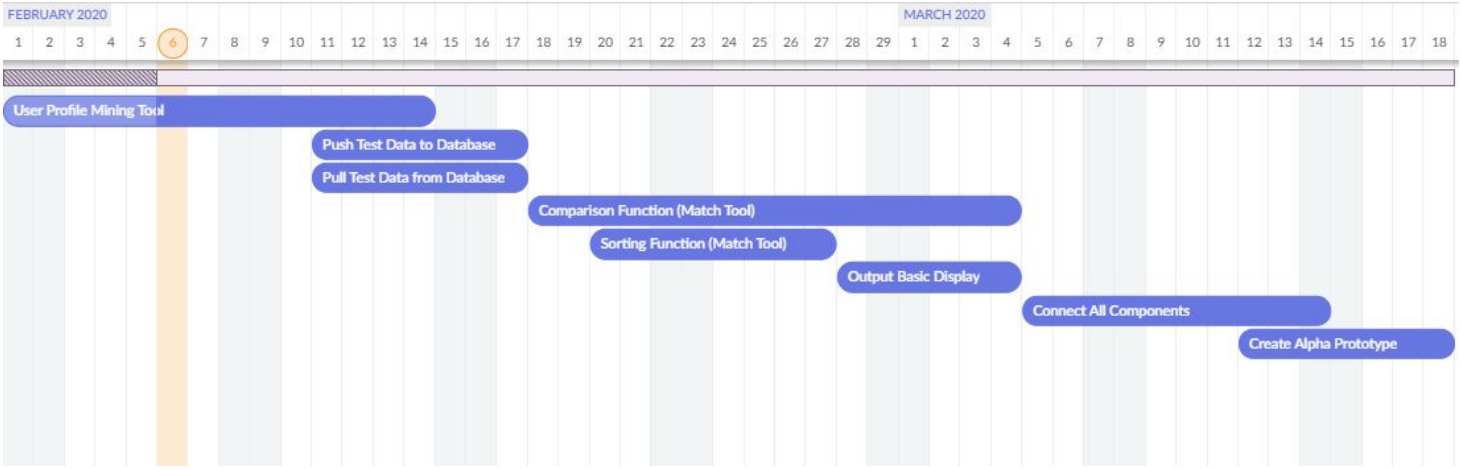

Figure 5: *A gantt chart showing our planned Spring 2020 semester schedule*

During the Fall 2019 semester, our team developed a prototype to provide the client with proof of concept. The team chose to demo the User Profile Mining Tool. While it was not complete, progress was made which continued into the Spring 2020 semester. As can be seen above, the user profile mining tool is almost complete and needs to be connected to the database. After this, we can make more progress on the other components. The figure above shows the new layout of our scheduled tasks until we are ready with an alpha prototype. The following information describes why these tasks are important and why they come before or after each other.

- User Profile Mining Tool: This tool and how it will be created has been mentioned in section 4.3, above.
- Push/Pull Test Data to Database: In order to start creating our matching tool, we will first need to learn how to easily push and pull test data to and from the database.

- Comparison Function (Match Tool): After we are able to push and pull data, we can then begin to create some sort of weighted skills algorithm that will start matching users with projects. This tool is also mentioned in further detail in section 4.5.
- Sorting Function: This sorting function is going to be a basic sorting function/functions that will help the user sort their matches, and can be completed after we have displayed matches.
- Output Basic Display: For our alpha prototype, we don't need a polished U.I., so what we want is a basic output display from our matcher.
- Connect All Components / Create Alpha Prototype: When all pieces are completed, we will connect them to create our alpha prototype

Additionally, below is a list of which team members were assigned to what task on the schedule aforementioned. The checklist will be shown to our client so they stay up to date on the progress of our project. Also, if the client has questions about a specific component of our project, they will know who to ask. The schedule and list of assigned tasks may be subject to change as unforeseen circumstances may arise.

Assignment List

	Joseph	Jevin	Adriana	Ugo	Stavros
User Profile Mining Tool	✔				
Push/Pull Test Data				✔	✔
Comparison Tool	✔	✔			
Sorting Functions			✔	✔	✔
Output Basic Display		✔	✔	✔	
Connect All Components	✔	✔	✔	✔	✔
Create Alpha Prototype	✔	✔	✔	✔	✔

Assignment List: A table showing which tasks are assigned to each group member

6 Conclusion

In summary, our team will be building a web application for Dr. Igor Steinmacher in order to support his efforts of encouraging newcomers to contribute to open source software. This application will allow users to login using their GitHub account, so we can gather information about their coding abilities, and then present them with open source projects that match their skills. An additional questionnaire will be provided to add additional skills not reflected in an individual's profile as well as for users that have never contributed on GitHub. We hope that this application will ease the overload of information on GitHub and make it more manageable, thus encouraging developers to contribute. We are confident that it will be of great service to the OSS community in the long run.

This document thoroughly outlines our implementation plans, an overview of the architecture of our application, descriptions of modules and the interface, and a timeline of how we plan to implement these features throughout the semester. We are eager and motivated to provide the best product for our client to aid his goal of making the world of open source software more accessible to newcomers.

7 References

[1] OpenSource. What is open source?. 2019. Retrieved December 10, 2019, from: https://opensource.com/resources/what-open-source

[2] Gousios, Georgios; Vasilescu, Bogdan; Serebrenik, Alexander; Zaidman, Andy. "Lean GHTorrent: GitHub Data on Demand" (PDF). The Netherlands: Delft University of Technology & †Eindhoven University of Technology: 1. Retrieved November 12, 2019.