CodeCollab Final Report

CS 3704 – Project Milestone 4.2

John Nguyen	Max Lane	Patrick Walsh	Rushi Bhut
johnnguyen02@vt.edu	lmax20@vt.edu	walsh968@vt.edu	rushibhut@vt.edu

ABSTRACT

The following is our problem statement: Many times during the development process, software engineers require expertise on areas they are unfamiliar with. Within a company there exist experts in different fields who can assist with these issues. However, a software engineer doesn't know everyone within their organization who could help resolve their issue. This leads to the following important question: How can we help connect these readily available experts to coworkers who need assistance?

To solve our problem and address that question, we came up with a product called CodeCollab. CodeCollab is a platform which helps connect struggling software engineers to experts within their organization who are skilled in that field. To achieve this functionality, CodeCollab maintains a database of software engineers in a company and their skills. CodeCollab also includes a feature which automatically schedules a meeting between the two engineers.

INTRODUCTION

The core job function of Software Engineering is to write code. As a software engineer, it is highly likely that you run into issues with your code. This could include syntax errors, runtime errors, not knowing how to approach a problem, etc. There are a wide variety of technical issues that software engineers encounter daily. While there are resources available to solve these issues (e.g. Stack Overflow, ChatGPT, documentation), it is usually helpful to get direct help from another software engineer. Especially a software engineer who is skilled in the topic related to the issue.

So why don't software engineers reach out to another engineer when they run across an issue with their code? Well, software engineers do not always know every person in their company. Specifically, engineers who are skilled in the topic the issue is about. Because most companies have NDAs, software engineers cannot reach out to engineers outside of their organization. Engineers who are experts in that topic within their company most likely do exist, but struggling engineers just don't know how to find them. This is what CodeCollab helps with. Our platform helps connect struggling engineers within their company.

The key benefit of our solution is that it reduces time to solve technical issues. Overall, this is beneficial to the entire company. Less time for resolving issues means that software engineers can get more work done. Our product accomplishes this by pairing engineers with experts. As mentioned earlier, there are other online resources that software engineers can use for debugging issues. However, if their issue is very specific, they may have trouble finding an online resource that is helpful. Direct help from an expert results in quicker times to resolve issues.

MOTIVATING EXAMPLE

In a practical example demonstrating a use case of CodeCollab, suppose there is a software engineer named Bob. Bob is currently working on a full-stack web development project. Bob is working on the frontend of the website, and this is his first project where he works on the frontend of a website. His company uses Javascript, and specifically React.js, as a primary frontend technology. Bob has a question about integrating React.js with the backend of the website he is working on. He has tried searching online for a solution but could not find anything because the problem is specific to the backend technologies that his company uses. Bob also cannot ask anyone else on his team to help him because the other engineers are working on the backend and have no frontend experience. Therefore, Bob decides to use CodeCollab to find a software engineer within his company who has expertise with React.js. Bob submits a request form to CodeCollab describing the issue he is facing. CodeCollab extracts the keywords from his description and uses a matching algorithm to find experts. These experts are matched based on keywords and schedule availability with Bob. Once the experts are found, they are all displayed (along with their available meeting times) to Bob. Bob selects an expert, and a meeting between Bob and the other engineer is automatically created.

This example shows how CodeCollab is relevant to software engineers working together because of its end goal. The end goal of CodeCollab is to pair software engineers within a company together to help resolve issues. Our platform promotes software engineers to work together with their coworkers instead of resorting to other online resources. As mentioned before, a positive impact of this is the reduction in time that it takes to resolve technical issues.

RELATED WORK

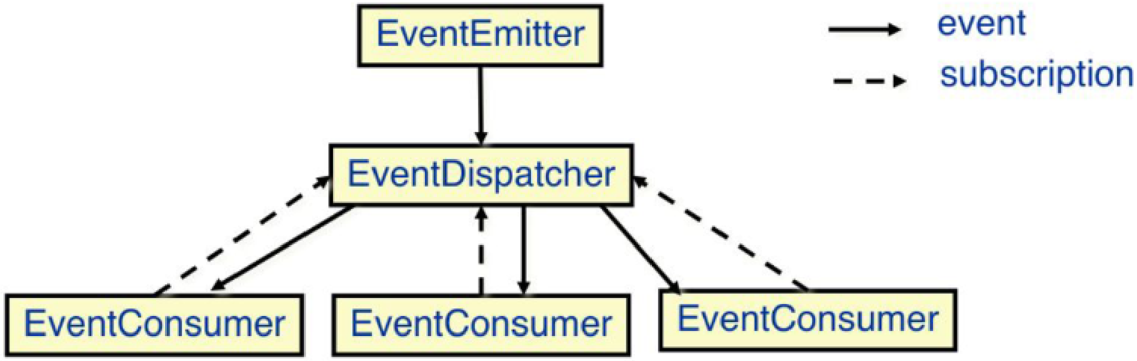
There are a few tools released that are similar to the idea behind Code Collab but don't exactly function the same or have the same features. Softwares such as Google Calendar and Microsoft Outlook both allow for multiple users or an organization to collectively share a calendar and schedule meeting but don't quite have the automatic meeting scheduler ability that is intended to be instilled in Code Collab. A different software that advertises as a "scheduling automation platform" is a software called Calendly. This software can, from what is being shown on its site, suggest meeting times based on user availability. Besides this there are some articles on the benefits of automating meeting scheduling as the article The Benefits of Automating Your Meeting Scheduling Process writes about how it can increase a company's or person's productivity, communication, and flexibility.

IMPLEMENTATION

The architectural pattern that we would use to structure our system is event-based architecture. From class lectures, we know that event-based architecture "promotes the production, detection, consumption of, and reaction to events." This perfectly describes our product (CodeCollab). Here is a brief review of CodeCollab: CodeCollab is a platform which allows software engineers within an organization to request assistance for issues relating to specific technologies. There is a database containing information about each software engineer and what technologies they are comfortable in assisting others with. When a software engineer requests for help through

CodeCollab, our platform pairs them with a software engineer who has expertise in that technology. Finally, a meeting is automatically created between the two software engineers based on time availability inputted by them.

As shown in the logical flow of our product, each step is based on the result of an event. For example, the first event would be a software engineer requesting help for an issue. The next event would be to locate another software engineer within the organization who has expertise with that issue (this depends on the result of the previous event). The following diagram from class can also be used to support the event-based architecture for CodeCollab:



In the example described above, the EventEmitter would be the creation of a request once a software engineer submits it. The EventDispatcher would be the search for another software engineer who can assist with the request. The EventConsumer would be the software engineer who is paired up to assist with the issue, and a meeting would be scheduled as a result.

The best pattern to structure CodeCollab is an event-based architecture because it can be divided into different events where each event depends on the result of the previous event.

DEPLOYMENT/MAINTENANCE PLAN

Since we have only four software engineers on the team, we have to perform all the development, operations, and maintenance. All of us will practice DevOps where we will have to automate as much as possible to reduce our workload and repetitive tasks. Our integration and deployment will be automated through Kubernetes where we will perform CI/CD. We will have monthly to six-month deployments where we will update and patch our software. To allow long gaps between each deployment, we will ensure maintainable code and reduce bugs by automated testing. UI testing through our licensed automated testing software will be run through every new iteration of our software. Unit tests will also catch any logical issues and code reviews between the team can improve readability. If any vulnerabilities and any other severe security issues get past our testing, we will provide hotfixes as soon as possible.

FUTURE WORK AND LIMITATIONS

Future work includes many other features to build a community and improve our algorithm, however, it may be hindered by the limitations of the project. Planned features are skill endorsements and ratings, collaboration tools, code review and feedback features, and a reward system. The majority of these features are to improve cooperation between the developers and incentivize experts to provide help to junior developers. There are also plans to improve the matching algorithm through machine learning based on skill endorsements and each meeting. However, the project is stunted by the lack of human resources and expertise. Four software engineers with barely any industry experience will find it difficult to implement these future features. Each deployment will provide a large gap between updates, but the workload may prove too much and may delay deployments and spill work over from one sprint to the next.

CONCLUSION

In conclusion, CodeCollab addresses the issue of connecting available experts to junior developers by facilitating a pipeline between them. This software will reduce the time and effort spent by companies to train, assist, and share knowledge with employees. In terms of project work completed, the project proposal has been submitted and accepted. Design, requirements, and testing plans have also been completed.

REFERENCES

(1) https://www.scrum.org/resources/what-scrum-module

(2) https://www.codepath.org/career-services/mentorship

(3) https://www.fiverr.com/

(4) https://www.letsmeet.network/post/the-benefits-of-automating-your-meeting-scheduling-process