Final Report: The PomBot

Isha Devgan, Stuti Shah, Michael Kopczynski, Anushka Trivedi

December 2024

1 Repository

Repository containing all project milestones and a README for this project can be found at this link: `https://github.com/idevgan/pombot`

2 Abstract

Software developers often face challenges in maintaining focus and preserving mental well-being due to poor time management and disorganized work-break patterns. The Pomodoro time management technique presents a structured solution to these issues by segmenting work into intervals followed by brief breaks. This paper introduces the Pomodoro Bot, a tool designed to help developers implement this method effortlessly. The bot offers customized Pomodoro cycles, allows for the adjustment of task durations and break intervals, and features capabilities such as team focus sessions and screen blocks during breaks. The goal of the Pomodoro Bot is to boost developers' productivity while promoting a healthier work-life balance through the establishment of consistent work and rest habits.

3 Introduction

Modern software development can be an intense field that demands sustained cognitive effort and meticulous attention to detail. However, many developers find it difficult to maintain extended periods of focus due to distractions, poor time management, and a lack of structured approaches to organizing their work and breaks. These obstacles can result in decreased productivity, burnout, and negative impacts on mental well-being.

The Pomodoro technique, a well-established time management strategy, effectively addresses these challenges by dividing work into focused intervals called "Pomodoros." Each Pomodoro typically consists of 25 minutes of focused work followed by a short break, with a longer break taken after four intervals. This method aims to improve concentration while providing regular opportunities

for mental recharging. Despite its proven advantages, consistently implementing the Pomodoro technique can be difficult without the appropriate tools to support and motivate users.

To address this gap, we propose the creation of a Pomodoro Bot specifically designed for the distinct needs of software developers. This bot not only automates Pomodoro cycles, but also offers customizable features that cater to diverse work schedules and project requirements. Furthermore, it provides real-time notifications, task tracking capabilities, and data-driven insights to help users assess their productivity patterns. Distinctive features such as team focus sessions enable groups of developers to coordinate their Pomodoro intervals for enhanced collaborative productivity, while screen blocking during breaks encourages individuals to step away from their devices and recharge. Using this tool, developers can overcome the challenges of effective time management, fostering a work environment that promotes focus and well-being.

4 High-Level Design

From the beginning of it's design, we envisioned PomBot as a Chrome Web extension. This would allow it to be seamlessly used by clients, as they would simply have to go to the Chrome Web store to download the application. On top of PomBot supporting user-controlled focus sessions, we wanted users to have the ability to customize their focus and break times according to their work needs. Furthermore, we wanted a built in task manager for users to assign corresponding tasks to their respective focus sessions. To encourage collaboration within teams, PomBot would have the ability to host group focus sessions so that teams are able to manage focus sessions and complete work and stay on the same page while doing so. Overall, these are some of the key features are the building blocks to making PomBot a unique productivity tool for software engineers.

5 Implementation

While implementing our project, we used a wide range of tried and true implementation strategies. For requirements elicitation, we used surveys and interviews to scope out features potential users would want within PomBot. The main design architecture we planned on using is a 2-layer Client-Server Architecture, allowing the client to communicate directly with the backend server while the server keeps track of data such as user information, and focus sessions.

6 Testing

Our testing strategy for the PomBot involves a range of testing techniques such as integration testing, UI testing, and functional tests. Our integration testing focuses on ensuring that PomBot's collaborate features allow for collaboration

among teams while maintaining PomBot's core functionality during individual use. Tests of this category would include stress tests on large group sessions and ensuring Gmail API integration. Our UI tests ensure that the UI is properly displayed, the timer system maintains a discreet profile, and no features are inaccessible or obstructive while using PomBot. For example, one of our UI tests ensures the button to start a focus session appears on screen and successfully starts a focus session. The functional tests ensure the app's core functionality is executed as intended. Among these tests are ensuring a user is able to edit focus session duration, and confirming a user is able to add a corresponding task to a focus session.

7 Deployment and Maintenance

In the software development lifecycle, designing and testing your program is only the beginning. In order to deliver a working program to users, software must not only be deployed but also continuously maintained. This section outlines the strategies our team would utilize to successfully deploy and maintain the PomBot application.

7.1 Deploying the program

As a Chrome Web Extension, the deployment of PomBot is restricted by Chrome guidelines. Although we cannot take as many liberties as independent applications, there are key deployment strategies we can utilize to ensure successful deployment.

7.1.1 Testing in Production-like Environments

As the development of PomBot is in its final stages, it is essential to ensure a bug-free initial deployment. However, we cannot deploy our extension to the Chrome Web Store without releasing it to the wider public, leaving little opportunity for pre-deployment user testing. To address this, we can utilize staging, the process of testing a deployment candidate in a production-like environment.

In terms of PomBot, this would entail testing the candidate in a Chrome or Chrome-adjacent environment. The candidate will be rigorously tested before deployment to ensure smooth user experience upon release. This strategy ensures that while we are restricted to basic deployment, we can deploy the best version of our program.

7.1.2 Canary Deployment

Once we believe our initial release candidate is ready for user testing, we can employ Canary Deployment before public release. Canary Deployment is the process of releasing an application incrementally to a subset of users in phases.

Since Chrome does not natively support this deployment style itself, we would privately distribute PomBot to hand-selected users. Once these users

return positive feedback, we can expand the pool of private users until we can confidently deploy PomBot to the public Chrome Web Store. This 'safety-net' minimizes risk and allows us to address potential issues proactively, ensuring a stable public release.

7.2 Maintaining the program

After deployment, continuous maintenance is essential to patch bugs, enhance functionality, and keep PomBot relevant. This involves a combination of iterative updating and advanced testing strategies.

7.2.1 Continuous Delivery

Once our PomBot application is live on the Chrome Web Store, we will begin the process of continuous delivery. Continuous Delivery focuses on building and releasing small, incremental updates that can be deployed quickly and reliably.

In our case, this would entail utilizing Chrome's automatic update system to easily deliver new features, bug fixes, and performance improvements. By following CI/CD principles, we ensure that updates are reliable and consistent to enhance user and developer experience.

7.2.2 Advanced Testing

As Pombot is updated and grows to a larger scale, our previous testing strategies would be insufficient or fallible in ensuring PomBot's core requirements are met. To resolve this, we would utilize the advanced testing strategy to validate the application while it undergoes changes in scale and functionality.

For example, our functionality tests largely rely on unit testing, which must be updated to accommodate changes in internal logic. To supplement these, we would employ fuzz testing, which involves intentionally introducing invalid or unexpected inputs to identify vulnerabilities.

Paired with frameworks that can automatically run these tests, we can ensure that PomBot is able to develop further without excessive time spent updating tests. These techniques would be fundamental for maintaining the application.

8 Related Work

Prior to beginning our work this semester, our team performed a market analysis to find relevant studies and popular tools to study software developers' attention spans and tracking time.

8.1 Relevant Research

To better understand the foundations of PomBot and explain its purpose, our team looked at recent studies on effective time management, attention, and mental effort, particularly in the field of software development.

Paper 1: Understanding Effort Regulation: Comparing 'Pomodoro' Breaks and Self Regulated Breaks (2023) [1] In this study, researchers examined how mental effort and job completion were affected by self-regulated versus systematic pre-planned pauses, such as those of the Pomodoro technique. Unsurprisingly, the research revealed that pre-determined breaks were a stronger method for time management and was conducive to reduced mental effort and an increase in task completion rate. Based on the findings of this paper, we decided to use the Pomodoro technique as the foundation of our time management tool.

Paper 2: Attention and Concentration for Software Developers (2023) [2] This paper outlined a study done to assess the attention spans of software developers in agile development settings using electroencephalogram (EEG) data. Using the EEG results, the study concluded that work environments that are focused on managing distractions and optimizing work conditions can help individuals maintain their focus. Thus, the PomBot, a tool that can be used in a work environment to reduce distractions, offers good support for individuals trying to concentrate for long periods of time.

By grounding our design in scientific research, we can highlight the direct issues that the PomBot aims to address.

8.2 Relevant Tools

After delving into the scientific attention span research, we wanted to identify existing solutions that address productivity in the market. We not only did this for inspiration, but also to identify gaps where our product could address unmet needs and provide unique value.

Toggl Track[3]: This tool helps individuals to efficiently monitor their work hours. It allows for automatic time tracking and reporting. However, Toggle Track primarily focuses on tracking time rather than managing it through break systematic work and break intervals. Additionally, this tool is often used as a company wide tracking tool for employees. This can be seen as disruptive and demotivate employees.

monday.com[4]: This tool is a cloud-based platform that provides users the ability to track tasks, be notified on all tasks, and create custom workflows. This tools inspired us to create task notifications for our PomBot. However, like Toggl Track, this tool does not offer structured focus sessions and break management.

After analyzing these tools, we identified how the PomBot can add to the already saturated time tracking tool market by combining time management through structured breaks, task notification, and distraction reduction for software developers.

9 Discussion

9.1 Limitations

While our team is confident in the PomBot application and its time management capabilities, we recognize that our project has certain limitations. For one, the tool relies on user adherence to scheduled breaks. Many developers like to work in a zone called the flow state, in which "you are totally absorbed by and deeply focused on something, beyond the point of distraction"[5]. Developers may want to adhere to this way of working rather than work breaks, and so the PomBot cannot suit all work styles. Additionally, the PomBot primarily addresses time management, but does not mitigate external distractions (not a feasible addition to our solution). Thus, we cannot completely understand the effectiveness of our product, as different work environments can lead to different distractions.

9.2 Future Work

Time was a main limitation to our project. However, our team planned future work and directions of our project. In future iterations of the project, we would like to add the possibility to have Pomodoro sessions that are pre-made, so a user doesn't need to manually configure sessions settings every focus session. We would also like to expand this project beyond a Chrome Extension in favor of a desktop application. Finally, our team will explore possibilities of PomBot integration with company specific software, such as having the PomBot in Teams meetings.

10 Conclusion

For this semester project, we identified a critical challenge faced by software developers: "Software developers often struggle to maintain sustained focus and mental well-being due to ineffective time management and unstructured work-break patterns." To address this problem, we developed a tool that promotes productivity and reduces mental effort spent on tasks through the Pomodoro technique.

As for project work completed, we conducted market and research analysis to identify a gaps in the market that the PomBot can address. We then completed the requirements engineering process through surveying potential users and create use cases. We created a high and low level design framework along with a wireframe mockup to depict our idea. Finally, for deployment and maintenance, our team developed a black box test plan for the PomBot.

We would like to implement our design and deploy it for testing and feedback on the project.

11 References

[1] F. Biwer et al., "Understanding effort regulation: Comparing 'Pomodoro' breaks and self-regulated breaks," Br. J. Educ. Psychol., vol. 93, suppl. 2, pp. 353-367, 2023, doi: 10.1111/bjep.12593.

[2] R. Amirova, G. Dlamini, A. Repryntseva, G. Succi, and H. Tarasau, "Attention and concentration for software developers," IEEE Access, vol. PP, pp. 1-1, Jan. 2023, doi: 10.1109/ACCESS.2023.3309414.

[3] Toggl Track, "Time tracking software for teams and freelancers," Toggl, [Online]. Available: `https://toggl.com/`. (Accessed: 12-16-2024).

[4] monday.com, "Work management software for teams," monday.com, [Online]. Available: `https://monday.com/`. (Accessed: 12-16-2024).

[5] Headspace, "What is flow state?," Headspace, [Online]. Available: `https://www.headspace.com/articles/flow-state`. (Accessed: 12-16-2024).