FocusBOT Final Report

Elise Turka
author
Virginia Tech
Computer Science

Brice Harris
author
Virginia Tech
Computer Science

Sanjana Ghanta
author
Virginia Tech
Computer Science

Ruhi Jame
author
Virginia Tech
Computer Science

***Abstract*—Workplace productivity is critical to meeting deadlines and ensuring smooth workflows. The FocusBot addresses key challenges in task management by offering a centralized platform that integrates daily task tracking, inactivity alerts, and progress reporting for employees and managers. By providing reminders for overdue tasks, visualizing progress, and enabling managers to receive end-of-day summaries, FocusBot enhances productivity and focus in corporate settings. These features address inefficiencies in traditional methods of tracking progress and improve both individual performance and overall team efficiency.**

1. Introduction

Dedication, discipline, and talent are all desirable qualities for potential hires in the corporate world. Consistency in productivity is what allows meetings to flow and deadlines to be met. Without hard-working people, companies would cease to make progress. This stagnancy could result in a multitude of negative outcomes such as bankruptcy or layoffs. Keeping employees on track from workday to workday is extremely important for ensuring companies are running efficiently. Oftentimes, there is not a unified process for tracking an individual's taskings throughout the day. Providing a corporate tool to support productivity will not only facilitate these desirable "new hire" qualities in all employees, it will also increase overall company production to levels not yet seen.

The following scenario will shed more light on the use of one such productivity tool. Imagine an employee of a company is not meeting their daily task expectations. Throughout the day, they take many breaks and often become distracted. The team manager is too busy to meet with the employee every day, so their declining productivity goes largely unnoticed. Now imagine that there was a tool specifically designed for this employee to add and modify tasks to a desktop task list. This task list is ever-present on the employee's screen to make it easier to remember what they need to accomplish. Imagine this tool also provides a noise signal when the employee is taking too-long or too-frequent breaks to remind them to log back on and continue working. Lastly, imagine the manager was sent a "progress report" of all people reporting to them; this would make it much easier for the manager to assess an individual's productivity and host meetings with them accordingly. This streamlined method to track progress, both for the employee and the manager, is the basis of FocusBot.

2. Related Work

In the age of the internet, there are many tools like the FocusBOT that help users with task management and productivity.

One productivity application already on the market is Microsoft's To Do application, which is a task management tool that allows users to manage their tasks [1]. This tool allows user to also create to do lists and share those lists with others as well as seamlessly integrating the lists and tasks into other Microsoft products such as Outlook. While this application works well for small, individual projects, it will not work as well for large scale projects. The To Do application is strictly a task management application, so it can only provide a list of tasks with deadlines. One review of the app states that "Microsoft's app did exactly what I'd want a to-do app to do. It just always seemed to be intent on making me take an extra step or two" [2]. Other sites claim the amount of options associated with Microsoft To Do can be overwhelming at times. When desiging the user interface for FocusBot, these considerations were taken into account and the app made as simplistic as possible.

Another productivity application is Habitica, which gamifies productivity and habit tracking [3]. Habitica allows users to stay on track by implementing instant gratification for completing a task in the form of earning points. The flow of the site goes like this: a user will create habits that they want to instill or accomplish. Once a task is completed and the user checks it off, their virtual avatar will level up and unlock features such as battle armor, pets, and quests. Finally, users can battle monsters with their powerful avatars or use their reward gold to watch an episode of their favorite television show. This style of habit tracker is extremely interesting because it really integrates users into their own personal games. One review of the app states that Habitica's only drawback is its lack of direct customer support [4]. Otherwise, it is a great tool to begin tracking good and bad habits. For the purposes of FocusBot, features that incorporated some sort of fun aspect were carefully

considered and chosen. These features include confetti when 100% of tasks have been completed as well as the user's ability to cater the aesthetic scheme of the app to their liking.

Pomodoro Timers are a productivity tool that helps users stay on task and focused for 25 minutes. Once the timer goes off, users are allowed to take a small break before returning to work [5]. These timers are an excellent way to help people that are easily distracted to fully focus for a dedicated amount of time. Additionally, taking regular breaks throughout the study session can allow the brain to reset and refocus for the next study period. While there are many benefits to using a Pomodoro timer for studying, there are also some drawbacks as well. These timers are heavily structured; interrupting an efficient study period with a break may lead to distraction or an unwillingness to continue studying. These timers also do not allow for collaboration and are mostly designed for individual tasks and projects. Overall, FocusBot incorporates some of the benefits and avoids some of the drawbacks of Pomodoro timers by not forcing "on time" and "off time". Instead, users can work as they please, only receiving a reminder to continue working after 30 minutes of inactivity on the device has been reached.

3. Architecture, Implementation, and Testing

Throughout the design of this project, many decisions had to be made regarding the architecture and implementation of the application. In terms of architecture, FocusBot employs a layered Model-View-Controller (MVC) high-level design approach. The model layer manages data such as daily tasks, progress updates, and inactivity durations. The view layer delivers a user-friendly interface, including a desktop task list, progress bar, and inactivity notifications. The controller mediates user inputs (e.g., adding or updating tasks) and system outputs (e.g., generating manager reports). A centralized "data center" maintains employee-specific task data and progress records to ensure seamless integration across features.

FocusBot utilizes a low-level design approach of behavioral design patterns. This is due to the fact that behavioral design patterns provide a chain of responsibility that would deal with the objects. For example, a subtask of the application is the computer pinging a sound after 30 minutes of inactivity to keep users on track for their breaks. Additionally, behavioral design patterns utilize encapsulation in a manner that aligns well with FocusBot's main objectives.

The primary implementation features of FocusBot include: tasks lists, an inactivity timer, visual feedback, progress bars, and manager reports. The task list allows users to add, modify, and remove tasks, with updates reflecting immediately in the progress bar. Completed tasks trigger visual feedback, such as falling confetti when 100% of the tasks are checked off.

The inactivity timer monitors user activity and emits a sound after 30 minutes of inactivity. This is an almost backwards approach to the Pomodoro Timer, wherin 25 minutes of studying is followed by a pre-planned 5 minute break. Here, FocusBot is attempting to mitigate excess break time and ensure that employees are being productive throughout the workday.

The progress bar is simply a visual representation of the task list. The user has the opportunity to revise tasks that will be updated in tandem on the progress bar. Additionally, color, font, and sizing modifications may be made to the user's liking.

Finally, the manager view report is a special feature only given to team managers to track the overall progress of the team. The report is delivered to the manager with all employee names, employee ID's, task lists, and progress bars displayed for easy review.

Requirements were an extremely important part of the design of FocusBot. Both stakeholder and software developer's specific requirements were all noted and taken into account. Some of these requirements focused on the usability of FocusBot. To make it easy to see the progress bar and task list, any graphic must take up at least 10% of the screen size. To be reliable, the program must achieve a minimum of 7 days of mean time between failures. To improve performance, FocusBot must reflect any updates to the task list, progress bar, or inactivity timer within 1 millisecond of the update occurring. Finally, FocusBot must be available across a variety of operating systems, including but not limited to MacOS, Windows, and Linux.

Finally, in terms of implementation, the 10 Usability Heuristics of UI Design were a heavy influence for the formation of FocusBot. First, the "aesthetic and minimalist design" heuristic was incorporated into the look of the progress bar to avoid frills and excess information. The "user control and freedom" heuristic was incorporated by allowing the user to add, delete, and modify any existing or new task. The "error prevention" heuristic is utilized here by presenting an error message to the user when the due date for a task is not inputted correctly. Within the error message, the correct format is reminded to the user to hopefully prevent any additional errors. This feature also ensures that all tasks are accompanied by accurate due dates.

FocusBot's testing process includes unit, integration, and user acceptance testing (UAT) to ensure a reliable and user-friendly system. Unit testing focuses on verifying individual components, such as the task manager and inactivity timer, to confirm they function as intended. Integration testing ensures smooth communication between components, such as task completion updates correctly reflecting on the progress bar and in manager reports. Finally, UAT gathers feedback from diverse users to evaluate and refine the interface and functionality, addressing any gaps in usability. Automated tools simulate various user scenarios, while real-world testing ensures accurate inactivity detection and minimal delay in UI updates, meeting performance benchmarks.

4. Deployment

Software development for an entire program must take a holistic approach. Deploying and maintaining any software, whether it be a website or an update to Windows, requires thought as to the most efficient process. For FocusBot,

deployment will be initially geared toward the desktop version. Additional platforms for the software, such as an app version, may be made in the future.

For the desktop application, an agile process will be followed. Agile was chosen to be the main software engineering process because of its main principle to emphasize frequent delivery and continuous improvement. FocusBot has a long way to go and many areas where new ideas can enhance the software. The agile practice aligns with this issue. Agile also focuses on breaking down projects into phases. This principle will also assist with tackling one functionality at a time. For instance, fully coding the inactivity timer may take precedence over coding the graphics for the confetti associated with the progress bar. Due to its focus on frequent updates and breaking down assignments into chunks, agile will likely be the main process chosen to develop FocusBot.

When taking into consideration the SE process, type of system and usage, and the company/team initializing deployment, it became clear that a Blue-Green deployment system would make the most sense for FocusBot. Blue-Green deployment utilizes a blue color for staging and green color for production. Essentially, testing and validation are completed in blue while user traffic is shifted to green. The environments are allowed to switch after successful testing and deployment. Blue-Green deployment works well in conjunction with the agile software engineering process. Additionally, Blue-Green deployment is simple, easy to implement, and offers low risk. It is also straightforward for rollbacks. This deployment process will work well for FocusBot because some applications are able to be blocked and worked on while others can be left open for user traffic. The reason for not choosing another deployment tactic is that the team prioritized having FocusBot available to users at all times while also utilizing user feedback to more quickly identify problems with the software. While many deployment methods could apply to FocusBot, the Blue-Green tactic was the one that aligned the most with the requirements of the developers.

Maintaining FocusBot will go right along with the Blue-Green deployment method. Any part of the program that is currently being staged will be worked on efficiently. Once the bugs are resolved, the staged portion of the program will switch to being live to users and vice versa. The formerly live code will be assessed and any problems experienced by users during live time will be noted and worked through using a priority list. By using this process, FocusBot will be able to identify, mask, fix, and release issues with the program in a manner that benefits the users the most. The team will strive to make this software as bug-free and usable as it can be so that employees will be able to use it effectively to be more productive.

Throughout the semester, the team regularly used Kanban boards to keep track of progress and identify the next most important task. This trend will likely keep up during the development process for this program as it is a visual aid in task management for software engineering.

Finally, a dedicated team will likely be assigned for the upkeep of the app. Upon reading reviews about other task management applications, a common disappointment was their lack of customer service. To counteract this, a team will be set up to help aid users when the app goes down. This direct interaction will also allow for developers to clearly see issues with the existing program and strive to alter the way the app works to make it more user-friendly.

5. Discussion

FocusBot, while comprehensive in its applications, could still leave users wanting more in the way of features. This section aims to outline future work for developers when implementing FocusBot. Most evidently, the design of the interface needs to be refined. While preliminary sizes for graphics and outlines of employee reports have been chosen, coding of the GUI and color schemes must be chosen before the program can be released to the public. In conjunction with this, functionality for changing fonts/colors/placements and aesthetics must be designed to allow the user a balance between a good amount of choices, but not too many to become overwhelming.

In terms of additional functionality, incorporating push notifications to remind the user when tasks are near due or overdue would be an excellent addition to the program. The ideal times to send these notifications would be 1hr before due date, 30 mins before due date, and past due.

The last idea that would be beneficial to incorporate into FocusBot is a history report. This would function sort of like Spotify Wrapped in that it could give users a comprehensive view of their activity during the week, month, and year. Some statistics that may be interesting to include are time fluctuations when working, average time to complete a task, and tasks completed per day, month, or year.

Adding in these features to FocusBot would reinforce the program's usability and diverse target audience. When deciding which capabilities to include, extensive user feedback and stakeholder research must be done to optimize the features going in. Including too many features can be overwhelming for the user while not having enough will likely cause the program to fail. In the future, these considerations will require further research to deem which ones are worthwhile to include.

6. Conclusion

FocusBot addresses the critical need for improving workplace productivity by providing an intuitive platform for task management, progress tracking, and inactivity monitoring. By integrating features such as visual progress indicators, inactivity alerts, and end-of-day manager reports, FocusBot bridges the gap between employees and managers, reducing inefficiencies in traditional productivity methods. While the current implementation lays a strong foundation, future enhancements could include push notifications for approaching deadlines, history reports to analyze productivity trends, and customizable interface options. These additions

will further refine FocusBot's ability to support productivity, adaptability, and engagement in diverse corporate environments.

7. References

[1] Microsoft, 'Microsoft To Do', 2024, URL https://www.microsoft.com/en-us/microsoft-365/microsoft-to-do-list-app, accessed 10-December-2024

[2] ZDNet, 'Microsoft To Do review: A to-do list app that makes you do too much', 2022, URL https://www.zdnet.com/article/microsoft-to-do-review/, accessed 12-December-2024

[3] Habitica, 'Motivate Yourself to Achieve Your Goals', 2024, URL https://habitica.com/static/home, accessed 12-December-2024

[4] TechRadar, 'Habitica Helps You Keep Good Habits', 2022, URL https://www.techradar.com/reviews/habitica, accessed 10-December-2024

[4] Pomofocus, '25 Minute Timer', 2024, URL https://pomofocus.io/, accessed 1-December-2024