The World's Greatest TODO App

Team MJ'S

Moses Hughes
mahughes@vt.edu

Matthew Moore
mmoore00@vt.edu

ABSTRACT

Traditional to-do apps and checklists often do not satisfy the needs of project managers or software developers. They often lack flexibility or customization, leading to an inefficient software development plan. The World's Greatest TODO App offers improved customization, task management, and organization to software developers or project managers.

This is done through the development of systems supporting a teamwide TODO lists with claimable tasks, personal tasks with timelines and subtasks, integration with Google Calendar, projects that are made up of tasks, and managers being able to assign tasks to individuals within the team.

By increasing the amount of customization and organization options, The World's Greatest TODO app offers a way to improve the software engineering process by offering more organization and accountability for software developers.

INTRODUCTION

Traditional to-do apps often lack the necessary flexibility and customization for software development teams, leading to inefficient software development plans and inadequate organization.

Knowing the requirements is an important part of the software engineering process. According to *The CHAOS Report* by the Standish Group published in 1995, incomplete requirements and specifications was one of the top factors impairing projects. Conversely, some of the top reasons that helped project success was executive management support and clear requirements.

Since having a clear set of requirements is crucial to having a smoother software design cycle, The World's Greatest TODO App has several features that help improve the software development process. The app allows managers to assign different roles and groups, giving more organization for assigning tasks or showing the completion of tasks for a project. With this, teams are better able to collaborate with each other and other teams. The app also offers integration with other calendar apps, such as Apple iCloud calendar or Google Calendar, giving developers a visual timeline of when to complete tasks.

By allowing project managers to have an easier time organizing and assigning tasks to software developers with the TODO app, it helps set clear deadlines which in turn will help lead to more successful projects.

RELEVANT WORK

In a 2020 journal published by the International Research Journal of Modernization in Engineering Technology and Science, titled *A REVIEW OF DAILY PRODUCTIVITY GROWTH USING TODO MANAGER [1],* by Suyash Kejriwal, Vaibhav Vishal , Aastha Gulati and Gaurav Gambhir, TODO lists can be improved by adding several key features: partial task completion, task efficiency, and a target completion indicator.

A partial indicator would allow users to be able to complete smaller tasks, making management more efficient.

A task efficiency visualization would allow users to see how efficient they are by seeing completed and pending tasks, keeping accountability.

A target completion indicator would allow users to set a goal for themselves, compared with daily task efficiency, keeping users motivated.

The journal provides many improvements to improve TODO lists to help improve people's efficiency in completing tasks, thus improving the software engineering process.

A 2016 Journal published in the Journal of Graduate Medical Education titled *Getting More Done: Strategies to Increase Scholarly Productivity [3]*, by Sarina Schrager and Elizabeth Sadowski, discusses how making TODO lists have a biological impact on increasing work productivity. This is known as the "Zeigarnik Effect"; writing down tasks reduces the need for the brain to remember those tasks which means it can focus more on completing those tasks.

SOFTWARE ENGINEERING PROCESS

The Agile engineering process will be utilized in the development of The World's Greatest TODO App. As an iterative process, it provides great flexibility for the team to respond to any issues that arise, user feedback, and development of new features following an initial deployment of the app.

To do this, the team will design and then implement relatively simple goals or features in small time increments. For example, the team will start by developing a basic personal TODO list. This will give a good baseline to how the final app will work. From there, we have a few options on where we could go. From our preliminary planning, we could develop a team based system with multiple interlinking lists, add timelines to tasks and integration with third party calendar applications, or some different feature that we haven't thought of but users desire.

Alongside this, the team will conduct group surveys both within the team and with a selected outside group for group elicitation of the current state of the project and any additional features that should be added to the program.

DEPLOYMENT & MAINTENANCE PLAN

To deploy The World's Greatest TODO App, we will adhere to a comprehensive deployment plan, incorporating continuous integration practices and a canary deployment strategy. The process begins with rigorous pre-deployment activities, including code reviews, automated testing, and static code analysis to ensure code quality and security. Continuous integration will be employed, automating the build process and integrating various levels of testing into the pipeline. To do this, we will develop automated tests of the most commonly used and important features for the app. This will allow us to verify that any new features developed and implemented do not break already existing features. Artifacts will be securely stored for traceability.

The canary deployment strategy involves an initial deployment to a dedicated environment accessible only to our internal development team. This allows for thorough testing and identification of any issues. Upon successful internal testing, a gradual rollout to a small percentage of actual users will be initiated, followed by deployment to increasingly larger portions of the user base. Throughout this process, monitoring of production environments for anomalies, and collecting feedback from both internal and external users will be ongoing.

Following the successful canary deployment, the full rollout to the entire user base will take place. Monitoring will continue closely post-deployment.. Post-deployment activities include additional testing, documentation updates, and user communication to inform users of changes and new features. We intend to include group elicitation techniques at this point to determine the state of our product in the eyes of our users. An incident response plan will be in place to address any unforeseen issues promptly. This deployment plan aims to ensure a seamless release, prioritize user experience, and facilitate ongoing improvement through continuous monitoring and feedback.

Maintenance will follow the deployment indefinitely and include analysis of group elicitation results and development and implementation of findings.

HIGH LEVEL DESIGN

We use the Layer/Tiered Architecture to structure our project. More specifically, we decided the Model-View-Controller / 3-layered architecture would be best suited for our project. This is mainly due to the MVC architecture's inclusion of UI. Our project will be reliant on a well made UI system that can take various inputs to process. The application will then have to process these inputs to manipulate a database of all of the team members (users) and all of the tasks for the team. It will then have to access this database and update the UI to show the users the correct information about the tasks.

LOW LEVEL DESIGN

The Behavioral Design Pattern Family would probably be helpful for our project. Besides designing the UI, this project would revolve around two main types of objects, team-members and tasks.

State Pattern: Status patterns can be used to model the different states of a pending task (e.g., unfinished, completed, in progress). The behavior of a task can change based on its current state. For example, you can use different text colors to mark work in different states.
Template Method Pattern: The template method pattern can define a skeleton algorithm for managing tasks, and the specific steps are implemented by subclasses. For example, the interaction and task between user software and third-party software can be realized through subclasses.

Memento Pattern: Memento can be used to capture the status of a to-do list, allowing users to undo or redo changes. For example, the user's work requirements when creating a to-do event list are only approximate requirements and are not precise. It can be used to store the status of the list before and after modification, so that the user can adjust according to the process.

Strategy Pattern: Strategy patterns can be used to encapsulate different sequencing strategies for tasks. Users may want to sort tasks by deadline, leader requirement, priority, creation date, difficulty to overcome. The Strategy pattern encapsulates the algorithms, and makes them interchangeable. It lets the algorithm vary independently from clients that use it allows Easily switch between strategies

For our program, we would make the most use out of the State Pattern as our program relies on tasks being implemented, then having their "state" changed from "to-do", to "in progress", to "completed". Making use of the State Pattern allows us to accomplish this with our project.

REQUIREMENTS ANALYSIS

We will use a few methods to determine the requirements of our project. First, research will provide us with opinions of experts in the field on what is needed for an effective work management system. We will consider these requirements and compare them to our experience with other work management systems. Will also perform surveys to the same effect.

Once we have a working prototype of the app we plan to perform focus groups. This would allow us to get together software developers who would be using our software and can tell us, in their experience, what works and doesn't work for apps like ours. This will be extra helpful in our circumstance because as software developers they will have an understanding of how the app works and what is realistic to implement.

CONCLUSION

We decided to develop The World's Greatest TODO App to address some issues we had with existing team/task management systems. Our research brought us to the paper *A REVIEW OF DAILY PRODUCTIVITY GROWTH USING TODO MANAGER [1]*. This paper provided us with areas of focus for development of our app, namely features providing partial indicators, task efficiency visualization, and target completion indicators. These features included: having broad projects that are broken into tasks, of which can then be broken down further into subtasks; adding timelines to tasks and projects that can be exported to external calendar services like Google Calendar; and allowing both individual users and team supervisors the ability to add/modify tasks for a team.

To accomplish this project we decided to adopt an agile design process with focus on user feedback and interactions. Agile was chosen as the process due to its iterative nature, which allows us to develop the app with an immediate response to issues or additional features that are requested for the application.

At this time, T.W.G.T.A has not been completed. While we have worked on developing the inner-workings of the app, we have not developed any UI elements. Our biggest limitations at this point are the size of our team and our lack of a graphic designer or anyone proficient in UI/UX development. We also plan on implementing authentication and security systems in the future. The final piece of work remaining is performing our planned group elicitation. This is dependent on having a working prototype with an at least rudimentary UI developed. We will perform focus groups and surveys to determine user satisfaction with our product.

REFERENCES

[1]“A REVIEW OF DAILY PRODUCTIVITY GROWTH USING TODO MANAGER,” *IRJMETS*, vol. 2, no. 12, Dec. 2020.

[2] “The chaos report (1995) - California State University, Sacramento,” Sacramento State, https://www.csus.edu/indiv/r/rengstorffj/obe152-spring02/articles/standishchaos.pdf (accessed Sep. 24, 2023).

[3] S. Schrager and E. Sadowski, “Getting More Done: Strategies to Increase Scholarly Productivity,” *Journal of Graduate Medical Education*, vol. 8, no. 1, pp. 10–13, Feb. 2016.