IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

CREATING A VIRTUAL WORKSPACE FOR PERSONAL OR BUSINESS UPKEEP AND ORGANIZATION OF DATA USING AI

Prof Sinu Nambiar Guide,
Shubhadeep Adhikari*1, Muskan Kalra*2, Vaibhav Chavan*3, Siddhesh Sonawane*4 Students
Department of Computer Engineering
Dr. D.Y. Patil Institute of Technology Pimpri, Pune, India

Abstract: In our current work-from-home reality, finding the ideal workspace software to manage both professional and personal commitments can be a daunting task. Many existing software options on the market are technically complex and may not be user-friendly, especially for those new to the online world. Therefore, our primary focus is to create a tool with a good User Experience (UI/UX) that is easy to understand and operate for the personal user. Our ultimate goal is to integrate organization and automation using Artificial Intelligence. We aim to provide users with a structured approach to daily tasks and streamline their workflow. By bringing together all the necessary tools in a sleek, well-designed UI, we hope to achieve our main objective of creating a functional and user-friendly software solution.

***Index Terms* - to-do lists, computer technology, productivity software, task management, metadata, email, calendar events, machine-learning algorithms, usability, likeability, intelligence**

I. Introduction

Nowadays, individuals rely on various computer and non-computer technologies to manage their personal to-do lists. There are numerous applications available for to-do list management on desktop and handheld computing platforms alone, generating a whopping $6.6 billion in annual revenue. However, despite the availability of personal information management functionality in almost all office productivity software packages, to-do list management components are often too simplistic and do not offer enough context or advanced features to improve productivity significantly. Current task management solutions are relatively basic, allowing for task entry, viewing of incomplete and completed tasks, and marking of work as complete. Furthermore, most to-do list management software tools are linked to email and calendar tools, but they require time and effort to keep up to date. Our proposed system, Notable, aims to bridge this usability gap by introducing a dynamic, contextually-based task management system that employs machine-learning algorithms to account for environmental context and user behavior. In this paper, we will outline the system's design and development, discuss related work, and provide a qualitative evaluation of its usability, likeability, and intelligence. Notable will offer an easier-to-use solution for individuals and businesses to manage their data efficiently.

II. Assumption and Dependencies

Assumptions:

When creating the project plan, the following presumptions were made:

1) The users are inputting correct information when acknowledging the completion of the task.
2) The users are all accepting the task allocated to them, other than the ones who are absentees or already working on a task.
3) Post verification of the work done is also being conducted manually by the assigners.
4) Each team member is pre classified into their own category.

Dependencies:

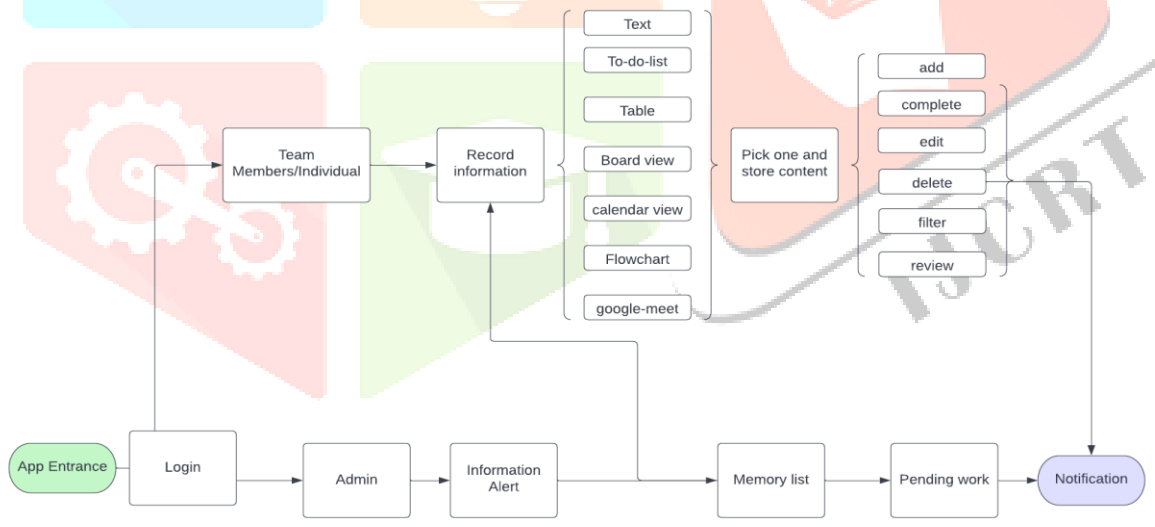
When creating the project plan, the following dependencies are needed:

1) *Bundle size*: Adding dependencies to the front-end bundle will increase the loading time of a page (more resources means more bytes to download means more time), and page loading time has a direct effect on conversion rate, so heavy and not optimized front-end libraries might become expensive to your business.
2) *External Libraries*: External libraries have given the ability to use code that was tested across many environments.

   External dependencies have 2 main effects on your web application:

   a. *Security*: Using external code could be dangerous since ambiguous libraries could add malicious code to your development environment or your user's runtime (intentionally or by supply chain attacks).
   b. *Performance*: Adding external code might cause some performance issues since using generic and solutions for relatively easy problems could cause an overhead. Also, some libraries "hide" bugs, that might cause memory leaks or CPU overuse when used in scale or untested environments. Some of these bugs are hard to detect, but as popular the open-source is, the less likely it will happen.

III. System Architecture



Our system architecture presents a comprehensive flow of the entire ecosystem. The user is provided with various options to create customized content in the Individual's Management section. Moreover, CRUD functionalities and the ability to modify the document's view are also available. On the other hand, the Team's Ecosystem is managed by the admin who can appoint other admins to manage the teams and team managers responsible for handling their respective teams. The team manager has the authority to assign tasks and update the task completion status.

IV. Methodology

Establish objectives:

1) Create a virtual workspace software for individuals and organizations with advanced features.
2) Connect different teams within an organization to improve workflow.
3) Integrate a Suggestion generator to enhance transcribing speed.
4) Allow users to create projects and tasks.
5) Implement an automatic validation detection system.

Algorithms:

Through our research, we have identified several algorithms that we can use at different stages of our project.

1) The Oracle Thesaurus Management System (TMS) can be used to delegate coding and QA duties to members of an organization.
2) Manual Pool Allocation is one way to allocate tasks. Omissions, unapproved VTAs, and Action assignments are accumulated in a pool and allocated to other TMS users manually.
3) Automatic Pool Allocation is another way to allocate tasks. Tasks accumulate in a pool until a process is invoked to allocate them to other users. You can select an algorithm that takes into account capacity or capacity plus current workload.
4) Direct Allocation is a third way to allocate tasks. TMS allocates omissions, unapproved VTAs, and unapproved Action assignments to users as they are created, based on user capacities that you define.

TMS directly allocates tasks in certain situations*:*

1) These situations include omissions that enter TMS through Batch Validation and terms with associated tasks that enter TMS through Disconnected System Integration (DSI).
2) TMS also directly allocates tasks on VTAs that are created by manually classifying an omission where VTA approval is required and on VTAs that are created proactively.
3) Additionally, TMS directly allocates actions assigned to omissions where approval is required.

Sorting algorithms are instructions that arrange items in an array or list into a specific order*:*

1) They are essential in computer science and have direct applications in various algorithms and data structures.
2) The most common sorting algorithms include selection sort, bubble sort, and insertion sort.
3) Selection sort repeatedly finds the minimum element from the unsorted part of the array and places it in the sorted part.
4) Bubble sort works by swapping adjacent elements if they are in the wrong order.
5) Insertion sort works by virtually splitting the array into a sorted and an unsorted part and placing values from the unsorted part at the correct position in the sorted part.

During implementation, several challenges may arise, including:

- Task validity, which refers to ensuring that tasks assigned to team members are relevant, up-to-date, and aligned with project goals.
- Virtual communication can lag, making it difficult for team members to communicate effectively and collaborate in real-time.
- Potential employee burnout can occur due to the high-pressure environment of the project, long hours, and excessive workload.
- The complexities of managing exponentially larger teams can make it difficult to ensure that everyone is on the same page, working towards the same goals, and meeting their deadlines.

V. Conclusion

Using K-means and Clustering we were able to prioritize the tasks given by the assigner. Once we got the sorted tasks, we use Decision Trees to maintain the hierarchy and SVM in order to classify the teams. We then use Automatic Pooled Allocation to divide each task to their specific cluster team. The correctness of the work is checked using Unfulfillable Assignment and then sorted using Bubble Sort. Certain prerequisite validation can be automated, done using a Selenium and the system will then recommend the next task to their team using Naïve Bias.

VI. Future Work

Future plans include impementing a more user-friendly web application to maximize productivity for consumers, simplifying the backend complexities using high-level algorithms, producing text files of meetings for each discussion in every team member during summary meetings, and prioritizing work according to urgency and employee potential.

VII. Acknowledgement

We would like to express our sincere gratitude to all those who have contributed to the successful completion of this research paper on our web application Notable that we developed using Django. Firstly, we would like to acknowledge our project mentor Mrs. Sinu Nambiar, for providing us with valuable guidance, support, and feedback throughout the project. Her continuous encouragement, timely suggestions, and constructive criticism have been instrumental in shaping the direction of this research. We would also like to thank our colleagues and team members who have contributed to the development of this web application. Their hard work, expertise, and dedication have helped us to achieve our goals. Furthermore, we would like to extend our appreciation to the open-source community for their invaluable contributions to the Django framework, without which this project would not have been possible. Lastly, we would like to express our gratitude to our families for their unwavering support and understanding throughout this endeavor. Their encouragement and patience have been a constant source of motivation for us. Thank you all for your contributions and support towards the completion of this research paper

References

[1] Evaluation of Asana, Odoo, and ProjectLibre Project Management Tools using the OSSpal Methodology: Joana Ferreira Marques and Jorge Bernardino.
[2] Smartphone to-do list application to improve workflow in an intensive care unit: a superiority quasi-experimental study: Mathieu Esposito, Pierre-Louis Rocq, Emmanuel Novy, Thomas Remen, Marie-Reine Losser, Philippe Guerci.
[3] Cyber-physical systems architectures for industrial internet of things applications in Industry 4.0: Diego G.S. Pivoto, Luiz F.F. de Almeida, Rodrigo da Rosa Righi, Joel J.P.C Rodrigues, Alexandre Baratella Lugli, Antonio M. Alberti.
[4] User Interface (UI) Design and User Experience Questionnaire (UEQ) Evaluation of a To-Do List Mobile Application to Support Day-To-Day Life of Older Adults: Di Zhu, Dahua Wang, Ruonam Huang, Yuchen Jing, Li Qioo, Wei Liu.
[5] TaskMinder: A Context- and User-Aware To-do List Management System: Brian M. Landry, Rahul Nair, Zach Pousman, Manas Tungare.
[6] Interpreter, a free Tool for Segmenting, Labelling and Transcribing Speech: Claude Barras , Edouard Geoffrois , Zhibiao Wu and Mark Liberma.
[7] Social media-based app organizing daily events: Erenis Ramadani; Agon Memeti; Florinda Imeri; Nexhibe Sejfuli-Ramadani; Florim Idrizi.
[8] Towel: Towards an Intelligent To-Do List: Kenneth Conley, James Carpenter.