British Journal of Computer, Networking and Information Technology
ISSN: 2689-5315
Volume 9, Issue 2, 2026 (pp. 20-34)
www.abjournals.org

DESIGN AND IMPLEMENTATION OF AN AI-POWERED TASK MANAGEMENT SYSTEM (TASKWISE)

Olawunmi Asake Adebanjo[1], Adebowale Oluwasegun[2], Anuriam Isaac[3], Adewuyi Joseph Oluwaseyi[4*], and Agoha Emmanuel[5].

[1,2,3,5]Department of Software Engineering, School of Computing, Babcock University, Ilishan-Remo, Ogun State, Nigeria.

[4]Department of Computer Science, School of Computing, Babcock University, Ilishan-Remo, Ogun State, Nigeria.

[*]Corresponding Author's Email: adewuyij@babcock.edu.ng

Cite this article:

Olawunmi Asake Adebanjo, Adebowale Oluwasegun, Anuriam Isaac, Adewuyi Joseph Oluwaseyi, Agoha Emmanuel (2026), Design and Implementation of an AI-Powered Task Management System (TaskWise). British Journal of Computer, Networking and Information Technology 9(2), 20-34. DOI: 10.52589/BJCNIT-LA6S3NNL

Manuscript History

Received: 17 Apr 2026

Accepted: 20 May 2026

Published: 19 Jun 2026

ABSTRACT: *In today's fast-paced digital environment, individuals struggle to manage multiple personal, academic, and professional responsibilities efficiently. Most existing task management systems that we are used to using rely heavily on manual input and lack intelligent automation, natural language interaction, or personalized recommendations. These limitations result in fragmented workflows, reduced productivity, and inconsistent user engagement. This paper presents TaskWise, an AI-powered task management system designed to automate scheduling, enable natural language interaction, and provide personalized productivity assistance. The system integrates Natural Language Processing (NLP) and Firebase cloud infrastructure to support real-time task creation, intelligent reminders, and adaptive responses. The system architecture follows a three-tier model comprising a presentation layer built with React and Tailwind CSS, an application layer implemented client-side through Firebase SDK service modules, and a data layer using Firebase Firestore for real-time synchronization. Usability evaluation indicates that TaskWise improves task organization efficiency, reduces manual input time, and enhances user engagement through conversational interaction. By combining artificial intelligence, cloud synchronization, and intuitive design, TaskWise connects the functionality of traditional task managers with intelligent productivity assistance.*

KEYWORDS: Natural Language Processing; Task Automation; Firebase; Cloud Synchronization; AI Assistant; Productivity Systems.

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

British Journal of Computer, Networking and Information Technology
ISSN: 2689-5315
Volume 9, Issue 2, 2026 (pp. 20-34)

www.abjournals.org

INTRODUCTION

The use of technology for daily work is now a necessity, and in the process, it has changed the way we arrange our time. With the current fast-paced world, everyone finds themselves juggling with a large number of tasks both professionally and personally, necessitating efficient schedule-making and tracking. Mobile and online task management tools have made the process easier by making it possible to keep track of activities and monitor them efficiently. However, most of the applications available continue to rely on manual entry (Brynjolfsson et al., 2023). The development of Artificial Intelligence (AI) has resulted in a significant advancement that can be used to increase productivity by enabling reasoning and intelligent responses in the system. Using technologies like NLP and voice recognition, AI makes communication between humans and computers easy through the use of simple languages (Khurana et al., 2023). In this case, when one gives a command like 'Remind me of attending a meeting at 10 a.m. every Monday,' it automatically sets up and repeats the task (Noy & Zhang, 2023).

Task management systems like Trello, Todoist, and Microsoft To Do have been created to help individuals organize themselves better and manage their tasks efficiently (Trello Inc., 2025; Doist Ltd., 2025). Some of the basic features they provide include the ability to create to-do lists and set reminders, among others. Nevertheless, there is no feature provided in these applications that enables them to understand human language and recommend appropriate tasks (Doist Ltd., 2025).

The development of Artificial Intelligence (AI) has resulted in a significant advancement that can be used to increase productivity by enabling reasoning and intelligent responses in the system. Using technologies like NLP and voice recognition, AI makes communication between humans and computers easy through the use of simple languages (Khurana et al., 2023). In this case, when one gives a command like 'Remind me of attending a meeting at 10 a.m. every Monday,' it automatically sets up and repeats the task (Noy & Zhang, 2023).

Task management systems like Trello, Todoist, and Microsoft To Do have been created to help individuals organize themselves better and manage their tasks efficiently (Trello Inc., 2025; Doist Ltd., 2025). Some of the basic features they provide include the ability to create to-do lists and set reminders, among others. Nevertheless, there is no feature provided in these applications that enables them to understand human language and recommend appropriate tasks (Doist Ltd., 2025). The development of Artificial Intelligence (AI) has resulted in a significant advancement that can be used to increase productivity by enabling reasoning and intelligent responses in the system. Using technologies like NLP and voice recognition, AI makes communication between humans and computers easy through the use of simple languages (Khurana et al., 2023). In this case, when one gives a command like 'Remind me of attending a meeting at 10 a.m. every Monday,' it automatically sets up and repeats the task (Noy & Zhang, 2023).

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

Statement of the Problem

Individuals living in the current high-speed society are often confronted by difficulties in effectively managing different tasks related to their personal lives, education, and professional sphere. Despite the abundance of modern technologies, most of the currently available solutions for task management do not provide for automated processing or any sort of natural interaction between the user and the system. One is forced to jump from one application to another to check deadlines and other information.

To begin with, most mainstream and freely accessible task management applications lack seamless AI-based automation capable of analyzing user intentions or generating schedules directly from natural language instructions (Hassan & Lim, 2023). Hence, it turns out to be a repetitive and time-consuming process of task creation and scheduling. Secondly, natural conversation and text-based interaction cannot be used in most currently available task management applications. As such, users have to navigate the app via strict menus (Jaddoh et al., 2024). This poses obstacles for those who prefer more natural modes of interaction.

Thirdly, numerous apps have standard dashboards that do not have any means of customization. In other words, there is no way to learn from the user's experience or recommend specific activities according to the analyzed patterns (Chun et al., 2025). Therefore, it is hard for the users to get motivated enough to follow the recommended pattern of managing tasks. Lastly, there are many productivity apps that collect information locally or asynchronously; hence, updates are provided with a delay (Shukla et al., 2023).

Objectives

The purpose of the current study is to design and build TaskWise, which is an artificial intelligence-driven tool for managing tasks using NLP and cloud synchronization features.. The objectives of this research include (i) thedesign of a reliable, easy-to-use, and responsive user interface for task management on both web and mobile devices; (ii) development of an AI text-based assistant with Firebase support for real-time synchronization, authentication, and data storage features; and (iii) testing the developed application to evaluate its performance and accuracy.

LITERATURE REVIEW

The fast-paced development of digital technology has brought about significant changes to the way people manage their personal and professional lives. The ability to manage tasks has changed from using pen and paper planners to managing personal and professional activities through advanced technological systems. Task Management Systems (TMS) have become important components in helping people improve time management skills and increase productivity (Kumar et al., 2025).

Task management software, including Todoist, Trello, Notion, and Microsoft To Do, has seen widespread use by millions of people in the past decade. Such software enables the creation of tasks, setting of reminders, progress monitoring, and synchronization with other devices. Despite being helpful in organizing activities and increasing productivity, there has been a

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

limitation regarding the intelligence and adaptation to individual needs of today's users (Kumar et al., 2025).

The adoption of AI technology in task management software is one of the most significant milestones in the digital productivity environment. The technologies include natural language understanding, scheduling, task prediction and reminders. NLP enables the task management software to convert human interaction into executable tasks. It is evident that through AI adoption in digital productivity applications, the users' experience is highly enhanced compared to conventional methods of task creation (Khurana et al., 2023).

Despite the benefits, there are still some difficulties in implementing these task management systems integrated with artificial intelligence technology. The reasons for not using these systems in developing countries are the high illiteracy rate, absence of stable internet connections, and lack of knowledge about smart devices. Another reason why these systems are not widely used is that most productivity tools require subscriptions, which makes them expensive and unaffordable for poor users (Okafor & Ibrahim, 2024). The implementation of productivity applications gained much popularity in Nigeria since the emergence of the coronavirus disease because people started using innovative methods of conducting their work and studies at home (Omodan, 2021)

Overview of Existing Systems

Todoist is a cloud-based application used to manage tasks with characteristics such as simplicity, repetitive tasks, and cross-device synchronization. However, the tool does not offer extensive AI automation and collaboration tools (Features Review: Todoist, 2025). Trello is a visual task manager designed to facilitate teamwork and project management using the Kanban method. While being intuitive, it can easily be confusing when dealing with larger projects and lacks intelligent automation (Trello vs. Asana: Productivity Review, 2025). Microsoft To Do is a productivity application that comes included with the Microsoft Office suite, offering features like reminders and sub-tasks but still lacking in automation and customization. Asana is designed for teamwork and process management and excels at integrating with cloud applications, but lacks features for conversational AI and automation. ClickUp is highly advanced in terms of goal setting and automation; however, it remains difficult to use. Google Tasks is a simple tool linked to Google Mail and Calendar but does not provide the ability to categorize tasks, analyze work, or automate processes. Trello is a visual task manager designed to facilitate teamwork and project management using the Kanban method. While being intuitive, it can easily be confusing when dealing with larger projects and lacks intelligent automation (Trello vs. Asana: Productivity Review, 2025). It is a productivity application that comes included with the Microsoft Office suite, where it offers features like reminders and sub-tasks but is still lacking in automation and customization.

For Asana, which is made for teamwork and process management, it is great at integrating with cloud applications, but there are no features for conversational AI and automation. ClickUp is highly advanced in terms of goal setting and automation; however, it is still difficult to use. Google Tasks is an easy tool linked to Google Mail and Calendar but does not provide the ability to categorize tasks, analyze work, or automate processes. All the other applications have certain features, yet they suffer from a number of similar problems, such as limited automation via artificial intelligence, scattered features, minimal personalization options, and expensive subscription prices.

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

Gap Analysis

Although there have been significant strides made in digital task management software as well as the inclusion of artificial intelligence in such systems, there are still some notable gaps that exist. In particular, current task management systems provide only rigid task lists that do not personalize themselves based on user behavior, prioritization, and productivity style (Gadhvi et al., 2025; Carrera-Rivera et al., 2024; Li et al., 2024). Furthermore, current task management and note-taking solutions are very fragmented, making it necessary for users to switch between different applications for scheduling, reminders, note saving, and setting goals (Lopez & Singh, 2021; Mark et al., 2005). Although today's software is becoming more automated, very few take advantage of AI-powered automation using natural language processing capabilities.

Many areas around the world are faced with infrastructural challenges like high data costs and lack of device capability (Oyelere et al., 2023; Okafor & Ibrahim, 2024). Many task management applications require external services like file storage and synchronization that raise privacy concerns (Global Data Privacy Trends, 2025). Most productivity applications do not support natural human-computer interaction through dialogue or voice control features (Abbas & Lee, 2024). These gaps highlight the need for a system like TaskWise, which addresses fragmented workflows, limited AI automation, and the absence of conversational interaction within the constraints of available connectivity infrastructure.

SYSTEM ANALYSIS AND DESIGN METHODOLOGY

The design was achieved through the use of the Incremental Software Development Model (ISDM), which makes it possible for the development process to be broken down into small, testable stages based on feedback from users at each stage. This model is particularly suited to the modular design of TaskWise.

System Architecture

The architecture of the TaskWise system is constructed using the concept of three tiers: the Presentation Layer, the Application Layer, and Data Layer. Such a tiered architecture improves the scalability, maintenance, and security of the system. Figure 1 presents an overview of the entire architecture of the system.

The Presentation Layer creates the user interface for interacting with the system from different devices such as web browsers and smartphones. It is built using React (TypeScript) and Tailwind CSS to create a responsive, stylish, and accessible interface. The Presentation Layer is responsible for accepting text-based natural language input and changing the theme between dark and light modes. The Application Layer contains all business rules and AI operations that are responsible for authentication, automation of tasks, voice and text processing, and real-time data synchronization. In particular,the Application Layer utilizes Firebase SDK service modules to handle business logic and AI APIs for natural language parsing and automated task suggestions.. The Data Layer is responsible for storing and retrieving data securely in the system using Firebase Firestore for cloud data storage and real-time transactions.

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

British Journal of Computer, Networking and Information Technology
ISSN: 2689-5315
Volume 9, Issue 2, 2026 (pp. 20-34)
www.abjournals.org

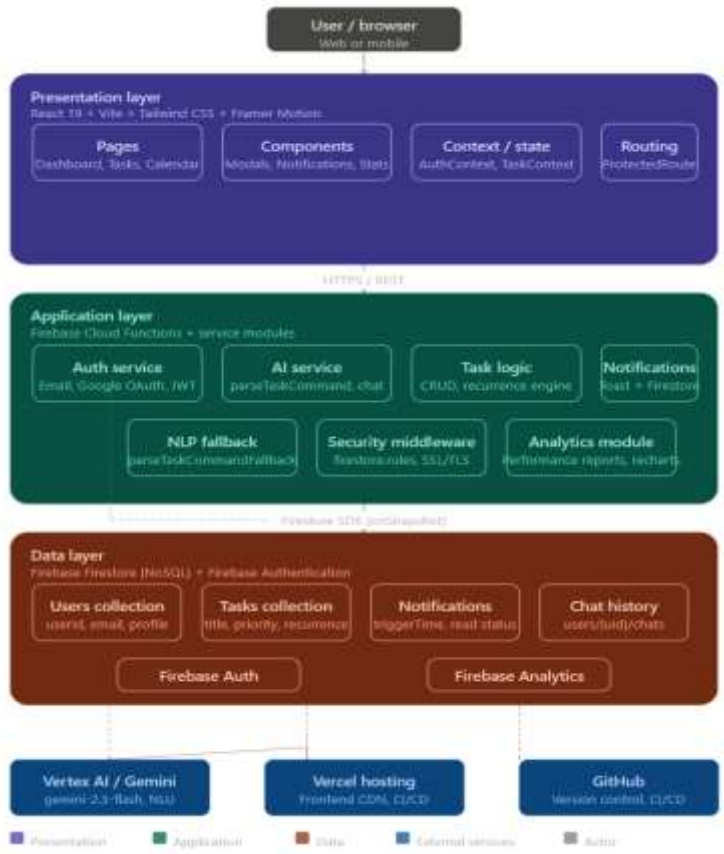
Tools and Technologies

For the frontend, React (TypeScript) technology was utilized for developing components that were both modular and type-safe in design, while the Tailwind CSS framework was employed for providing consistent styling. Firebase Hosting provided for deployment and updates in real time. In terms of backend operations, Firebase SDK service modules were utilized to manage authentication, data operations, and AI processing directly from the React frontend. For the storage of unstructured and structured data, Cloud Firestore offered real-time synchronization between user information, metadata of tasks, and settings of automations and logging, which ensured that updates on a single device would be visible simultaneously across other devices. Integration with AI took place through NLP functionalities that enabled users to perform tasks via natural commands.

Implementation Phases

During Phase 1, requirements, both functional and non-functional were gathered and recorded. In Phase 2, UI/UX designs were made using Canva and Figma tools. Phase 3 involved building the frontend using React and Tailwind CSS. Phase 4 consisted of constructing the backend using various Firebase services, where Firestore was used for data storage and Firebase Authentication was used for security purposes. Phase 5 included adding the AI assistant, which interprets natural language commands through Firebase's serverless function capabilities and third-party AI APIs to enable automated tasks, reminders, and contextually relevant recommendations. Phases 6 and 7 involved testing the app and deployment, respectively. Figure 2 illustrates the framework of the project and process.

Fig. 2: System architecture (Layered)



Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

Functional Requirements

The application enables users to sign up with valid email log-in details (FR001), uses Firebase Authentication to verify users' credentials with encryption (FR002), permits users to add, update, and delete task records complete with deadlines and categories (FR003), gives access to view all the tasks on their personal dashboards (FR004), uses artificial intelligence automation for task analysis (FR005), delivers real-time alerts for upcoming and reminder tasks (FR006), allows users to manage their profiles and upload photos on Firebase Storage (FR007), tracks completion rates and creates analytics for progress tracking (FR008), and updates preferences among several other devices instantly (FR009).

Ethical and Security Considerations

Users have complete control over their data, and there is no collection of sensitive data like financial details or biometric details. The AI assistant makes sure that the user knows that the suggestions are from an automated process, and the AI assistant does not take any personal data like gender, religion, or social status into account while making suggestions. Firebase Authentication and JWT are used for secure user authentication, and all sensitive data is transmitted securely using SSL/TLS and stored securely using Firebase internal encryption techniques. Firebase Firestore Security Rules make sure that proper access control policies are enforced and database backups are performed periodically using Firebase internal backup systems.

SYSTEM IMPLEMENTATION AND TESTING

The front end was built with React.js 19 and Vite, where an innovative composition matrix was used with Tailwind CSS and Framer Motion. Components are broken down into modularized sub-folders such as "components/dashboard" and "components/common." React Contexts such as TaskContext and AuthContext act as Controllers for the application that have perpetual synchronization loops monitoring document mutations in real-time via onSnapshot hooks. The back end is built on Google Cloud utilizing Firebase, where Firestore functions as the data Model while Cloud Rules act as the Middleware Layer.

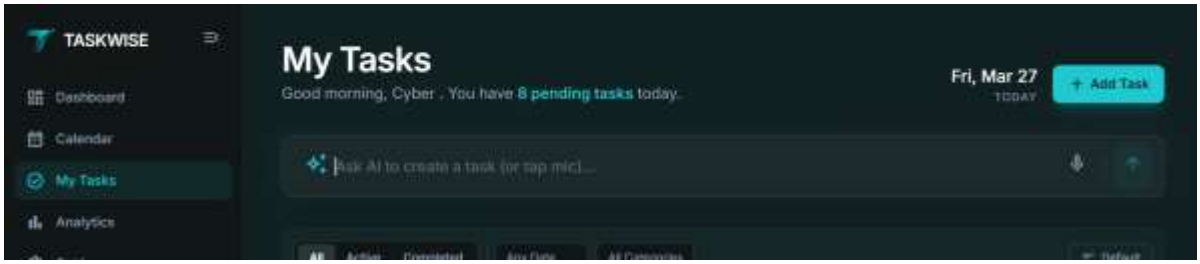
AI Integration

The initialization of the model employs getAI and VertexAIBackend. The AI service (aiService.js) employs a JSON mime type generation model to interpret natural language commands and convert them into task objects. The chat and planning methods generate natural language text or structured plans, where fallbacks have been programmed for situations where the AI service fails. The Intelligence Wrapper Layer creates instruction structures that incorporate complex limits to the persona and fallible fallbacks. In the event of failed network transactions for the gemini-2.5-flash model, the operation is directed to parseTaskCommandFallback() using native JavaScript string parsing to determine date, priority, and task variables.

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

British Journal of Computer, Networking and Information Technology
ISSN: 2689-5315
Volume 9, Issue 2, 2026 (pp. 20-34)
www.abjournals.org

Fig. 3: AI Assistant Interface

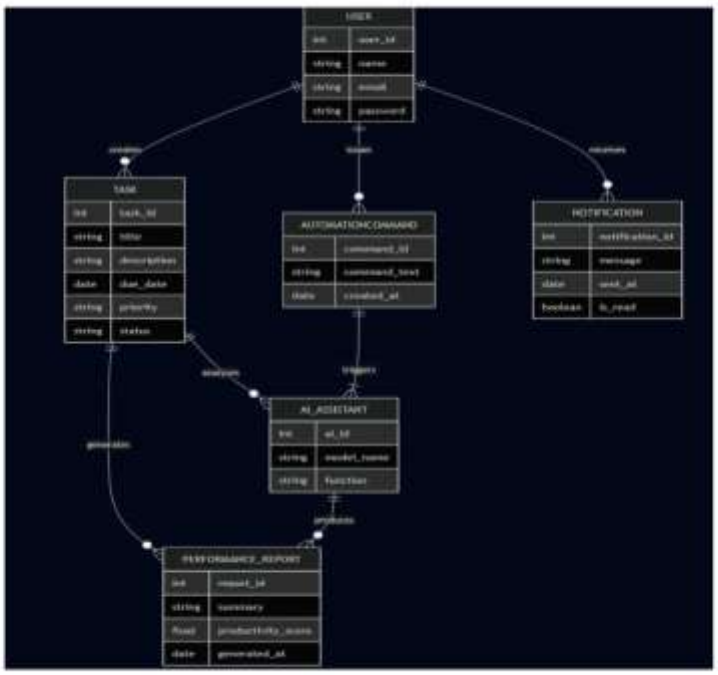


Database Design

The system follows a NoSQL schema, which is horizontally scalable. The Users collection (/users/{userId}) includes userId (String, Document ID), email, displayName, photoURL, and createdAt attributes. The Tasks collection (/tasks/{taskId}) comprises taskId (String, Document ID), userId (Reference FK), title, dueDate (ISO String), priority, recurrence (Map/Object), completed (Boolean), createdAt, and completedAt attributes. Access control is achieved through firestore.rules, wherein rules are executed for all incoming requests and verify whether each mutation parses through the conditional matrices in order to ensure that only authorized users access their own data. Figure 4 presents the Entity Relationship Diagram, and Figure 5 shows the Firestore Users collection with its document structure.

The Firestore Security Rules enforce strict access control, ensuring that authenticated users can only read and modify their own data. The rules validate authentication state and document ownership for every data operation. Figure 6 illustrates the consolidated Firestore Security and Database Rules configuration.

Fig. 4: Entity Relationship Diagram



Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

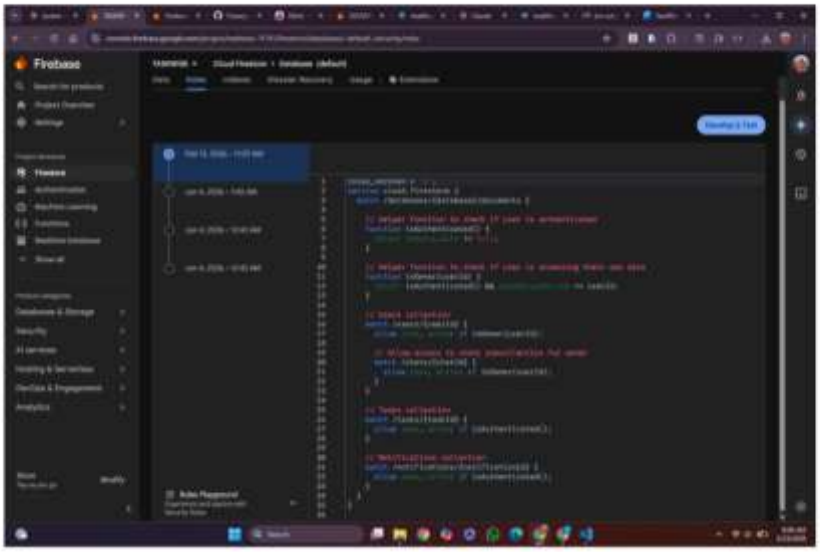
British Journal of Computer, Networking and Information Technology
ISSN: 2689-5315
Volume 9, Issue 2, 2026 (pp. 20-34)
www.abjournals.org

The database design also includes dedicated collections for user data and task management. Figure 5 shows the Firestore Users collection with document structure, and Figure 6 shows the Firestore Security Rules configuration.

Fig. 5: Firestore Users Collection

The Firestore Security Rules enforce strict access control, ensuring that authenticated users can only read and modify their own data. The rules validate authentication state and document ownership for every data operation.

Fig. 6: Firestore Security Rules



Functional Testing

The core functionalities were validated via test cases both manually and through scripting. The test case one was designed to validate user registration and login functionality, where it was confirmed that the newly registered users were able to access the dashboard. The test case two was meant to check, create, update and delete functionalities of the tasks that were successfully executed by persisting the results in Firestore. The third test case was developed to validate recurring task creation functionality, which was successful in adding new task documents along with their due dates calculated.

Usability Testing

The six participants who tested the tool included both students and peers, and they had to register themselves; create three different tasks, one of which should be recurring; use AI to generate a task; and make edits to their profile. On average, it took 28 seconds to generate a task (SD = 6.3s). On average, the SUS was calculated to be 82/100, which falls in the 'Good' usability range per Bangor et al. (2009). It should be noted that a sample size of n=6 is insufficient to draw statistically generalizable conclusions. These results are indicative only and should be validated through broader testing with a larger and more diverse participant

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

group in future evaluations. Within these constraints, the tool demonstrated good usability, and the AI assistant functionality was appreciated by all participants.

Performance Testing

Simulation-based read/write latencies were assessed using Firestore and Firebase console random checks. The write latency on Firestore was less than 200 milliseconds, while the read latency was less than 150 milliseconds in a light load condition. The browser DevTools, Firebase console requests log, and Lighthouse for page responsiveness were employed as the testing tools. The performance tests indicated that the application runs fast when only one user accesses it. In a case where multiple users accessed the system simultaneously, there was an increase in the latency period.

Security Testing

Authentication was tested with respect to both the email/password flow and Google OAuth. The rules for Firestore were tested with respect to enforcing the concept of document ownership, ensuring that only the user that owned the task could edit his/her task. Task creation input validation and sanitization of output content created by the AI model were also checked.

Implementation Challenges and Solutions

Various difficulties arose during the course of the implementation process. First of all, it was difficult to consistently parse ambiguously phrased natural language commands, but it became possible after fine-tuning parseTaskCommandFallback in aiService.js. The second difficulty that appeared was connected with Firestore security rules because of the danger of unauthentic requests, which was mitigated by improving firestore.rules through gradual testing with multiple users. Another issue was related to the calculation of dates of recurring tasks; therefore, calculateNextDueDate was implemented in TaskContext. jsx to handle all recurrences in one place and to transform dates to ISO strings.

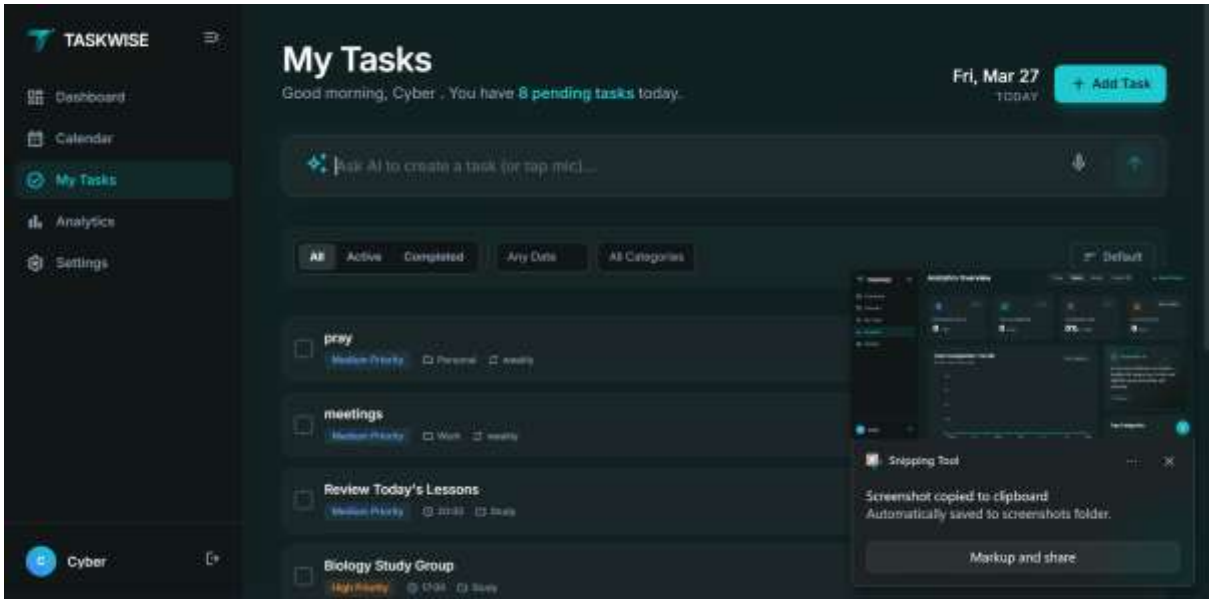
SUMMARY OF RESULTS

The functionality of all the major components like user login, task CRUD capabilities, scheduling, AI interpretation, chat logging, and notifications was working as expected on the prototype. Testing for usability gave us a great score of around 82 from SUS, while functional tests have also passed for the essential workflows. The performance is decent for prototype-level requirements, but extra work will be needed before deployment to production. The main idea of incorporating AI-powered automation and natural language processing within a productivity application has been proven successfully. The user interface screenshots can be seen in Figures 7 and 8 below.

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

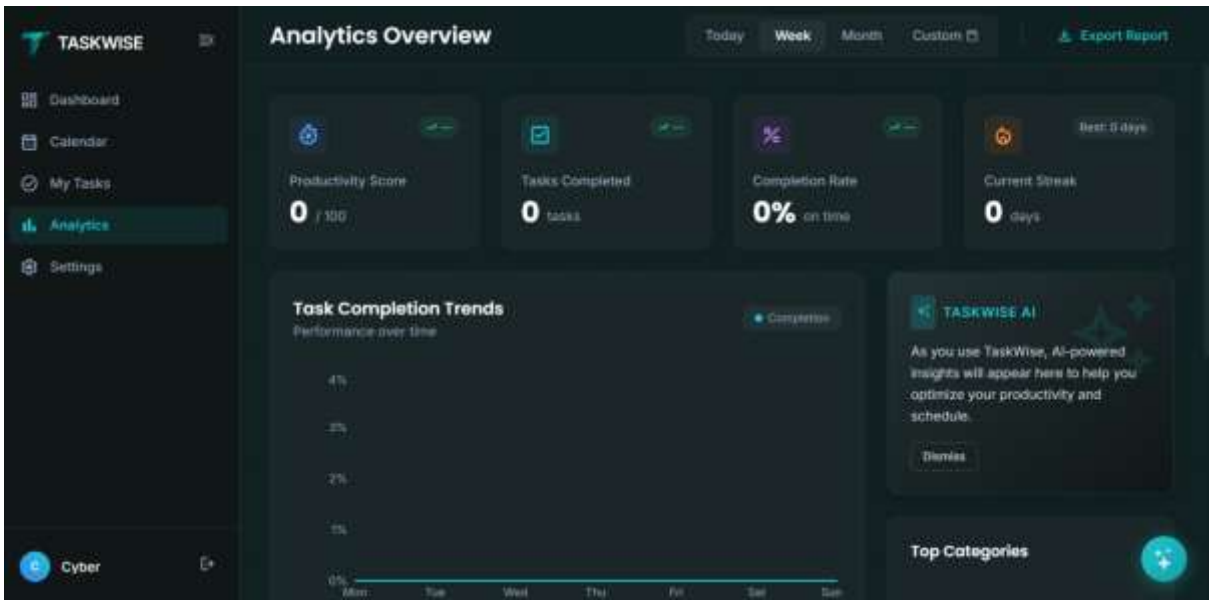
British Journal of Computer, Networking and Information Technology
ISSN: 2689-5315
Volume 9, Issue 2, 2026 (pp. 20-34)
www.abjournals.org

Fig. 7: Dashboard and Create Task Interface



The analytics dashboard provides users with visual insights into their task completion patterns, productivity trends, and time allocation across different categories.

Fig. 8: My Tasks and Analytics Dashboard



Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

British Journal of Computer, Networking and Information Technology
ISSN: 2689-5315
Volume 9, Issue 2, 2026 (pp. 20-34)
www.abjournals.org

Fig.9: Calendar

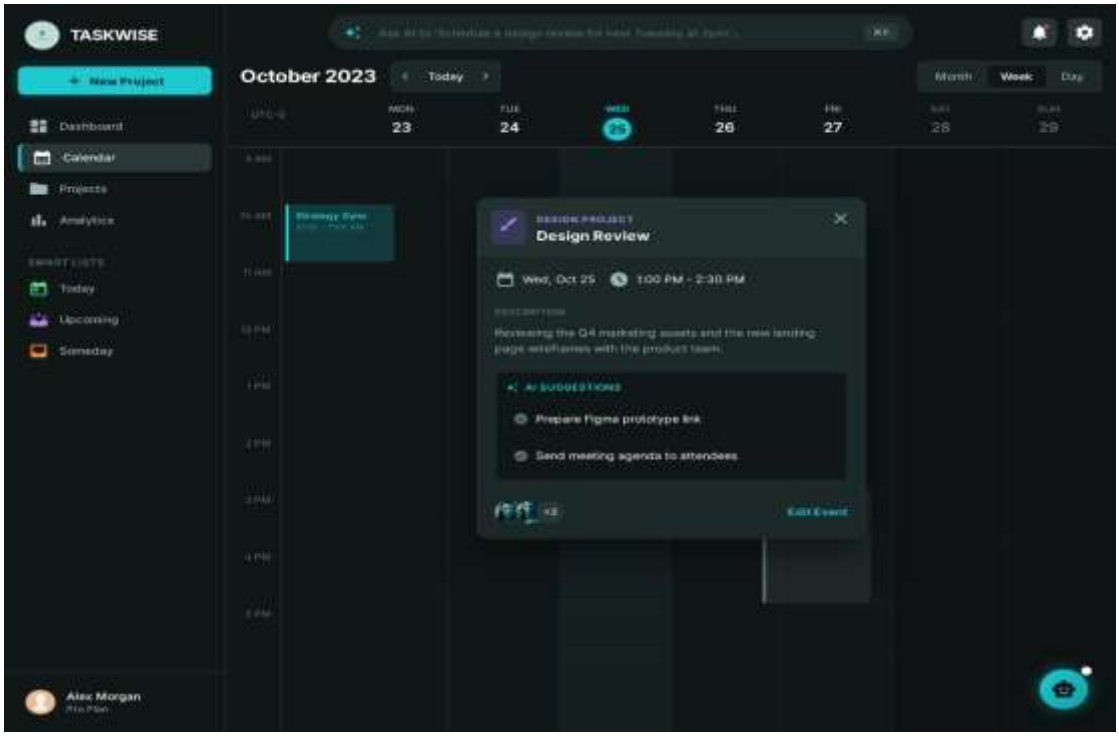
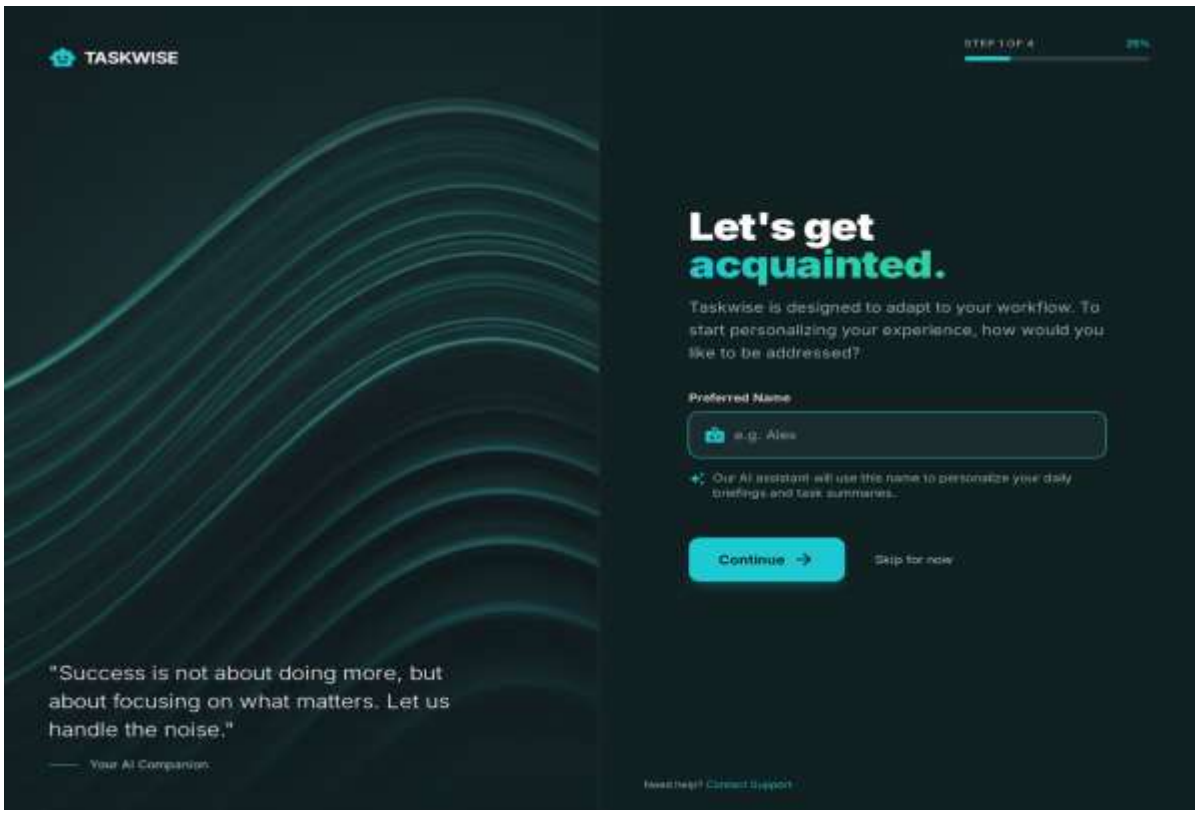


Fig. 10: Onboarding Page



Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

British Journal of Computer, Networking and Information Technology
ISSN: 2689-5315
Volume 9, Issue 2, 2026 (pp. 20-34)

www.abjournals.org

SUMMARY, CONCLUSION AND RECOMMENDATIONS

Summary of Findings

TaskWise is a successful project that involved the development of an AI-based task management software application that leverages natural language processing (NLP) and cloud-based synchronization to facilitate the automation of tasks. The implementation of TaskWise met its main goals through its capability to provide a safe, efficient, and intelligent task management platform via Firebase authentication, the Firestore database, and an AI-powered virtual assistant. The NLP engine effectively performed its role in the interpretation of instructions and transformation of conversation-based commands into scheduled tasks.

Contributions to Knowledge

The current study represents a foundational step toward implementing NLP-based task management in contexts relevant to developing countries; however, broader validation across more diverse and representative demographics is needed before claims of accessibility and usefulness to these populations can be substantiated. Moreover, it adds to the growing literature on synchronization architecture through the cloud using Firebase in an educational environment and thus can serve as a prototype for other developers. Finally, TaskWise creates the basis for further studies in developing AI-powered productivity apps in environments with limited bandwidth, giving empirical evidence of how AI productivity apps are accepted by users.

Limitations

The platform heavily utilizes the cloud resources provided by Firebase, and offline functionality has not yet been implemented; as such, TaskWise currently requires a stable internet connection and does not address the challenge of poor internet connectivity experienced in many developing regions. The capabilities of the AI assistant are limited to the use of rules and contextual information; it does not learn based on previous user experiences. Collaboration tools like workspace sharing are not available, and the current release is only compatible with the web version, with no apps yet developed for mobile phones. Additionally, this platform was tested on a small number of users, mostly college students, which limits the generalizability of the findings.

Recommendations

For future updates, it would be ideal if advanced machine learning models were employed to enhance natural language understanding; teamwork was added to allow task boards, scheduling, etc.; there could be development of mobile apps for both Android and iOS platforms, and integration with third-party calendar systems like Google Calendar, Microsoft Outlook, and Apple Calendar could be considered. Offline storage through the use of IndexedDB and Service Worker would be an excellent addition, and testing across varied demographics and education institutions is advised.

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

CONCLUSION

The TaskWise project is seen as a major advancement in using artificial intelligence for daily task management. TaskWise proved that automation through artificial intelligence and natural language processing can be effectively combined within one platform, which makes task management more productive. The tests conducted helped to confirm that the software was functional, user-friendly, and efficient. Although there were certain limitations associated with the prototype, the results of this work can serve as an important basis for further research in this field.

REFERENCES

[1] Abbas, A. & Lee, S. W. (2024). PITCH: Productivity and mental well-being coaching through daily conversational interaction. In Proceedings of Designing (with) AI for Wellbeing (CHI '24). ACM. https://doi.org/10.1145/nnnnnnn.nnnnnnn
[2] Allen, D. (2001). Getting Things Done: The Art of Stress-Free Productivity. Penguin Books.
[3] Al-Busaidi, K. A. & Al-Shihi, H. (2010). Instructors' acceptance of learning management systems: A theoretical framework. Communications of the IBIMA, 2010, 1-10.
[4] Alhabeeb, A. & Rowley, J. (2018). E-learning critical success factors. Computers & Education, 127, 1-15.
[5] Almaiah, M. A., Al-Khasawneh, A. & Althunibat, A. (2020). Exploring critical challenges and factors influencing e-learning system usage during COVID-19. Education and Information Technologies, 25(6), 5261-5280.
[6] Bond, M. (2024). A meta-systematic review of AI in higher education. International Journal of Educational Technology in Higher Education, 21(1), 36.
[7] Jobson, D. & Li, Y. (2024). Investigating the potential of using large language models for scheduling. In Proceedings of the 1st ACM International Conference on AI-Powered Software (AIware '24). ACM. https://doi.org/10.1145/3664646.3665084
[8] Chen, X., Liu, Y. & Zhang, H. (2023). AI-powered chatbots in learning management systems. Educational Technology Research and Development, 71(5), 2149-2167.
[9] Khurana, D., Koli, A., Khatter, K. & Singh, S. (2023). Natural language processing: State of the art, current trends and challenges. Multimedia Tools and Applications, 82, 3713–3744. https://doi.org/10.1007/s11042-022-13428-4
[10] Doist Ltd. (2025). Todoist Features Review. Retrieved from https://todoist.com
[11] Gao, L. & Fong, R. (2024). Gamification and task management systems. Behavior & Information Technology, 43(5), 892-910.
[12] Mark, G., Gonzalez, V. M. & Harris, J. (2005). No task left behind? Examining the nature of fragmented work. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '05). ACM. https://doi.org/10.1145/1054972.1055017
[13] Hassan, M. & Lim, J. (2023). Conversational automation in productivity tools. International Journal of Human-Computer Studies, 172, 102983.
[14] Noy, S. & Zhang, W. (2023). Experimental evidence on the productivity effects of generative artificial intelligence. Science, 381(6654), 187-192. https://doi.org/10.1126/science.adh2586
[15] Khurana, N., Majumdar, R. & Bose, S. (2020). A comparative study of mobile task managers. Journal of Mobile Computing, 8(2), 120-138.

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL

[16] Kumar, V., Sharma, A. & Singh, R. (2022). Integration of personal management features in learning management systems. Journal of Computing in Higher Education, 34(2), 233-249.
[17] Gadhvi, R., Petkar, S., Desai, P., Ramachandran, S. & Siddharth, S. (2025). AdaptAI: A personalized solution to sense your stress, fix your mess, and boost productivity. In Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25). ACM. https://doi.org/10.1145/3706599.3720284
[18] Li, M. & Ramos, J. (2023). Gamified productivity apps: Habitica and Forest. International Journal of Human-Computer Interaction, 39(12), 2341-2358.
[19] Lopez, M. & Singh, P. (2021). Team-based productivity platforms and collaboration. Journal of Collaborative Computing, 12(3), 178-195.
[20] Brynjolfsson, E., Li, D. & Raymond, L. R. (2023). Generative AI at work. National Bureau of Economic Research Working Paper No. 31161. https://doi.org/10.3386/w31161
[21] Mensah, J. & Boateng, R. (2024). Hybrid task managers: Manual control with AI suggestions. Information Systems Journal, 34(5), 1234-1252.
[22] Shukla, S., Hassan, M. F., Tran, D. C., Khan, M. K. & Debnath, N. C. (2023). Improving latency in Internet-of-Things and cloud computing for real-time data transmission: A systematic literature review. Cluster Computing, 26, 2657–2680. https://doi.org/10.1007/s10586-021-03279-3
[23] Jaddoh, A., Loizides, F., Rana, O. & Syed, Y. A. (2024). Interacting with smart virtual assistants for individuals with dysarthria: A comparative study on usability and user preferences. ACM Transactions on Accessible Computing. https://dl.acm.org/doi/10.1145/3726874
[24] Okafor, C. & Ibrahim, K. (2024). Subscription barriers and accessibility of productivity tools in Nigeria. African Journal of Information Systems, 16(2), 78-95.
[25] Oyelere, S. S., Suhonen, J., Wajiga, G. M. & Sutinen, E. (2023). Mobile-first productivity systems in Sub-Saharan Africa. Education and Information Technologies, 28(4), 1567-1585.
[26] Park, J., Kim, S. & Lee, D. (2024). Predictive analytics for task completion and user autonomy. Computers in Human Behavior Reports, 14, 100892.
[27] Chun, J., Zhao, Y., Chen, H. & Xia, M. (2025). PlanGlow: Personalized study planning with an explainable and controllable LLM-driven system. In Proceedings of the Twelfth ACM Conference on Learning @ Scale (L@S '25). ACM. https://doi.org/10.1145/3698205.3729541
[28] Omodan, B. I. (2021). COVID-19 pandemic and the new normal of remote work in Nigeria: Implications for employee productivity and job satisfaction. Journal of Contemporary Research, 7(2), 112-129.
[29] Carrera-Rivera, A., Larrinaga, F. & Lasa, G. (2024). AdaptUI: A framework for the development of adaptive user interfaces in smart product-service systems. User Modeling and User-Adapted Interaction. https://doi.org/10.1007/s11257-024-09414-0
[30] Li, N., Ban, X., Ling, C., Gao, C., Hu, L., Jiang, P., Gai, K., Li, Y. & Liao, Q. (2024). Modeling user fatigue for sequential recommendation. In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24). ACM. https://doi.org/10.1145/3626772.3657802
[31] Trello Inc. (2025). Trello vs. Asana: Productivity Review. Retrieved from https://trello.com
[32] Yao, S., Chen, L. & Zhang, W. (2022). Collaboration tools like ClickUp and Monday.com. Information and Software Technology, 145, 106809.
[33] Zhang, Y., Li, H. & Wang, J. (2023). Integrating personal and collaborative planning features. Journal of Software Engineering, 37(3), 456-473.

Article DOI: 10.52589/BJCNIT-LA6S3NNL
DOI URL: https://doi.org/10.52589/BJCNIT-LA6S3NNL