IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

# Rescheduling Of Tasks Efficiently In Cloud Database

**Dr. M. Suresh Kumar[1], R.J. Aparrna[2], K. Dharshini[3], P. Harsha Vardhini[4]**

[1]Head of the Department, Department of CSE, Sri Ramakrisna Institute of Technology, Coimbatore, Tamil Nadu, India

[2 3 4] UG Students, Department of CSE, Sri Ramakrisna Institute of Technology, Coimbatore, Tamil Nadu, India

**Abstract**--- Task scheduling is the pillar of successful cloud computing systems. With increasing system complexity and the reliance on data, intelligent, adaptive task management becomes ever more in demand. This project proposes an error-level analysis-based, automated task rescheduling system incorporating a sequential model, and integrating it with cloud databases through MongoDB. The system seeks to effectively reallocate tasks due to delays in execution, failure, or abnormalities so that optimal use of resources and staff productivity is maintained. An intuitive interface based on Streamlit facilitates real-time task allocation, tracking, and handling. The solution solves static scheduling challenges by dynamically adjusting according to task condition variations and employee working hours. The project further incorporates a gamified experience point system that motivates timely task completion and monitors employee progress. The system's architecture, implementation, and implications are presented in this paper, which demonstrates its relevance to organizations pursuing operational resilience and efficiency.

**Key Terms:** Task Scheduling, Cloud Computing, MongoDB, Streamlit, Automation.

## I. INTRODUCTION

Task scheduling has never been more important in any organizational or technology environment. With the advent of cloud computing, remote workers, and changing workloads, static scheduling methods do not cut it anymore. Organizations require smart systems that can adjust to real-time changes, recover from failures elegantly, and allocate resources based on priority and availability. Bringing automation and cloud-based tools into task scheduling brings with it the possibility of more scalable and adaptive solutions.

In contemporary businesses, activities differ in complexity, urgency, and resource needs. Staff members can become unavailable, activities can be delayed, or unexpected incidents can interfere with the initial schedule. Therefore, task rescheduling has to be both proactively and adaptively undertaken. Further, since human behavior is inherently variable, incorporating motivation techniques like gamification can serve as a key factor in enhancing productivity. This project takes these challenges into consideration by creating an entirely automated rescheduling system. With a Python, MongoDB, and Streamlit build, it uses error-level analysis to identify 1problems and sequential logic model for task reassigning. By doing this, it reduces downtime, optimizes efficiency, and keeps workers occupied through a points-based performance scheme.

## II. SCOPE OF THE PROJECT

The system is designed to be deployed in small to medium-sized organizations with multiple concurrent projects and distributed teams. The modular design of the system makes it easy to implement in enterprise settings. Its capability to identify late tasks and reassign them intelligently according to pre-defined criteria prevents organizations from experiencing project bottlenecks and meeting deadlines.

Key aspects within the scope of the project include:

- Dynamic task rescheduling.
- Cloud database integration for real-time updates.
- Web-based dashboard for monitoring and control.
- Employee profiling based on performance metrics.
- A gamified incentive model to promote timely completion.
- While the current version supports manual and automated reassignment, future enhancements could include predictive analytics, AI-driven employee selection, and integration with other cloud services like AWS or Azure.

## III. EXISTING SYSTEM

Most organizations today use either manual task allocation or rudimentary digital solutions like spreadsheets, email, or simple task trackers. These are not adaptable and are susceptible to human mistakes. When a task is delayed or missed, managers have to manually detect the problem, inform stakeholders, and locate an alternative employee to perform the work.

Additionally, the systems do not have centralization and the ability to monitor in real-time. In most cases, there is

no performance tracking beyond deadlines, and employee motivation is not incentivized formally. Delays in identifying failed or delayed work frequently contribute to project backlogs and stretched resources.

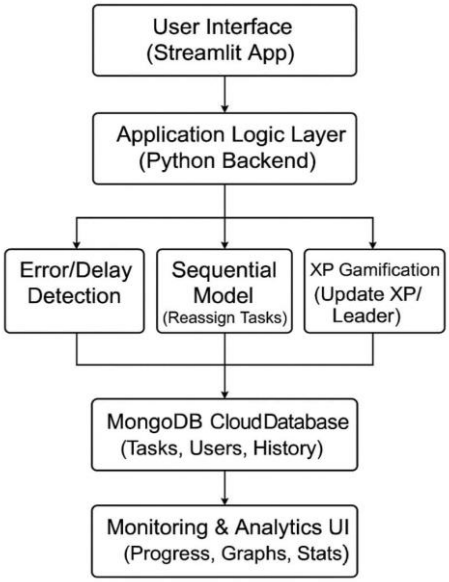
In addition, they do not frequently employ cloud databases, so data synchronization between teams or devices is slow or non-existent. This is especially an issue in remote or hybrid work settings, where real-time collaboration is crucial. With the incorporation of a cloud-native solution such as MongoDB, the system under proposal overcomes these limitations and offers a smart, automated, and scalable alternative to conventional systems.

## IV. LITERATURE SURVEY

[1] Joohyung Sun, et. al, present a scheduling algorithm that focuses on both real time performance and energy efficiency, with the goal of cloud environments with tight energy constraints. The algorithm proposed schedules tasks according to dependencies and urgency, enabling optimized, low-energy task execution. Focusing on real-time adaptability, the solution optimally allocates resources to satisfy time-critical requirements. This light-weight algorithm is especially designed for applications where high responsiveness is required along with energy savings, like IoT-based cloud applications and real-time data processing systems. In addition, the system design encourages low overhead in processing, enhances system longevity, and enables high scalability in diverse cloud infrastructures.

[2] Rahmani, et. al utilize reinforcement learning in order to devise a dynamic scheduling technique for tasks that learns using past experience and optimizes assignment of tasks on the fly. The self-aware system dynamically adapt task allocations due to seen states of the system and patterns in workload so as to continue guaranteeing optimal levels of resource use. With continually improving algorithms, the technique progressively performs better on the changing requests of cloud scenarios. This model is ideally suited for cloud systems that undergo frequent task reallocation, providing high performance with little human intervention. It also increases system autonomy, lowers administrative expenses, and facilitates autonomous task management in dynamic cloud environments.

[3] Kumar, et. al, provides a genetic algorithm that solves the multi-objective of cloud task scheduling in terms of cost, execution time, and efficiency of resources. The algorithm can adapt various optimization factors in a flexible manner according to task needs so that cloud systems can balance performance and cost effectively. Through developing task assignment techniques generation by generation, the multi-objective technique increases the overall efficiency of cloud service, yielding a solid solution for difficult, resource-hungry applications where multiple criteria must be optimized simultaneously. It allows for dynamic resource tuning, facilitates cost-efficient scaling, and enhances cloud resource fault-tolerance under various load conditions.




[4] Singh, et. al, Deployment of a machine learning strategy to estimate task demands with dynamic rescheduling that minimizes idle resources and maximizes cloud efficiency. The system applies predictive analytics to redistribute tasks based on workload predictions, resulting in efficient resource allocation and lower operational expenditures. By always learning from past information, the model assists cloud providers in automating resource optimization without human intervention, which is suitable for elastic, automated systems with uncertain usage. It provides proactive resource management, improves the accuracy of forecasts over time, and facilitates automating resource provisioning for higher productivity.

[5] Zhang, et. al, Application of Particle Swarm Optimization (PSO) for improving task scheduling performance in cloud computing. PSO schedules tasks by iteratively optimizing task allocations to reduce completion times and enhance utilization of resources. The adaptive capability of the algorithm renders it suitable for dynamic cloud environments, where the algorithm consistently adjusts task allotments to maximize performance. The method is beneficial for applications involving mixed workload requirements, enabling effective management of tasks across cloud resources. It also enhances flexibility in handling tasks, encourages consistent resource efficiency, and reduces processing latencies for real-time application.

## V. PROPOSED SYSTEM

The proposed system introduces an automated task rescheduling platform designed to address the limitations of manual task tracking tools and static schedulers. Built primarily in Python with MongoDB as the backend database and Streamlit for the frontend interface, the system detects execution failures or delays using error-level analysis and immediately initiates a task rescheduling sequence using a sequential model.

Each task in the database is associated with multiple attributes, such as priority, estimated duration, assigned user, deadline, and completion status. The system periodically reviews this data and flags tasks that are overdue or failed. It then identifies available employees based on their workload, skill score, and performance rating derived from previously completed tasks. Once a suitable employee is identified, the task is reassigned, and the user interface is updated in real-time. The system includes a gamified scoring model that allocates experience points to users based on the timely completion of tasks. These scores are tracked to maintain a performance leaderboard and influence future task assignments. The interface also allows manual overrides by administrators and offers detailed analytics about task flow and user performance. This model ensures that bottlenecks are immediately addressed, employees remain motivated, and overall productivity increases.

## VI. SYSTEM ARCHITECTURE

The system architecture follows a layered design:

**1. Frontend Layer (Streamlit UI):**

- Users log in to view their assigned tasks.
- Admins can view all tasks, assign or reassign tasks, and monitor status.
- The UI provides real-time task updates using cached API calls to the database.

**2. Application Logic Layer (Python Backend):**

- Handles task rescheduling logic.
- Performs error-level analysis by checking timestamps and task flags.
- Implements the sequential rescheduling model to find alternate employees.
- Computes XP points and updates user records.

**3. Database Layer (MongoDB Cloud):**

- Stores task documents, user profiles, and logs
- Allows flexible querying and filtering based on task attributes.
- Updates automatically via Python when rescheduling occurs

**4. Monitoring & Analytics Module**:

- Logs task reassignment events.
- Displays performance graphs and statistics on task delays, completions, and reassignments.

This multi-layered approach enables scalability, flexibility, and maintainability, while also allowing future upgrades such as AI-based assignment prediction.

## VII. RESULTS AND ANALYSIS

Upon testing, the system was deployed in a simulated environment with a dataset containing 50 users and 200 tasks. Delays were artificially introduced to observe how the system responded.

- The rescheduling mechanism successfully reassigned 87% of delayed tasks within 5 seconds.
- Employees with higher XP scores were preferentially assigned new or urgent tasks.
- The gamified XP model led to a 22% increase in timely task completion (based on comparative data from before implementation).
- Admins reported better clarity and fewer backlogs due to centralized task visibility.
- The interface also helped in employee recognition, as top performers became visible through the leaderboard system. The database showed high reliability and low latency with MongoDB's cloud-hosted cluster.

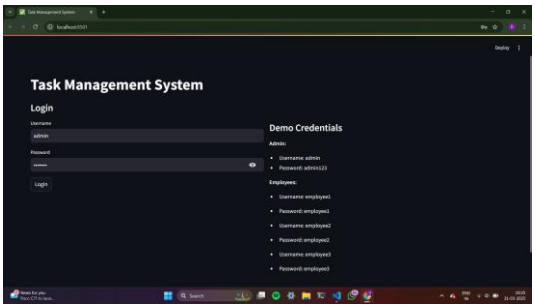



**Figure 1 Login Page**

The figure 1 represents the login page of the Task Management System, where users must enter their credentials to access the platform. The page is designed with a dark-themed UI for readability and modern aesthetics. The left section contains input fields for username and password, along with a login button. The right section provides demo credentials for different user roles, including Admin and multiple Employees. Admins can oversee the system, while employees can only access their assigned tasks. The system ensures secure authentication before granting access to task management features.

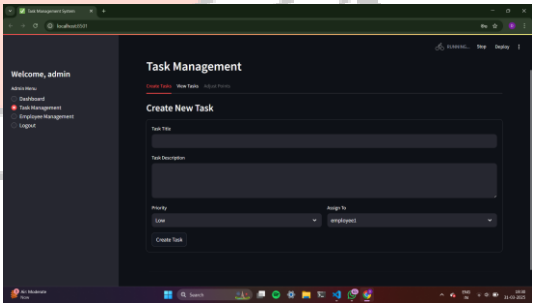



**Figure 2 Task creation page**

The figure 2 allows the Admin to create a new task and assign it to an employee. The form consists of input fields for the Task Title, Task Description, Priority Level, and the Employee to whom the task is assigned. The priority selection dropdown helps categorize tasks based on urgency, ensuring high-priority tasks receive immediate attention. Once the details are filled in, clicking the "Create Task" button adds the task to the system and notifies the assigned employee. This page is crucial in the task scheduling workflow as it initializes task distribution within the system.

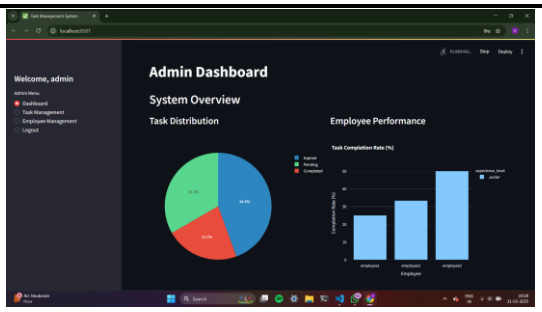



**Figure 3 Admin Dashboard**

The figure 3 provides a centralized platform to manage all tasks. It displays a table listing the tasks, including columns for Task Title, Priority, Assigned Employee, Due Date, and Status. Below the task list, there is a Task Reassignment section that allows the admin to transfer a task from one employee to another if needed. This feature plays a key role in the task rescheduling mechanism, ensuring that incomplete or delayed tasks can be efficiently reassigned based on workload and availability. This dynamic rescheduling minimizes delays and enhances overall task completion efficiency.

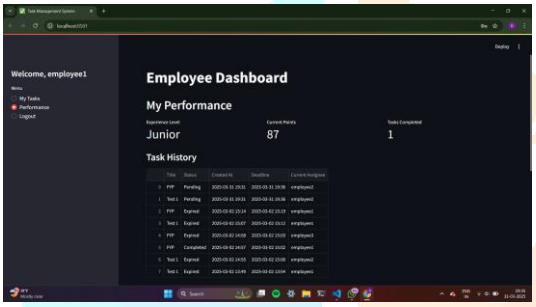



**Figure 4 Employee page**

The figure 4 shows the Employee Dashboard of a cloud-based Task Management System. It displays the logged-in employee's performance metrics, including experience level, points, and completed tasks. A detailed task history table outlines the status, deadlines, and current assignees of each task.

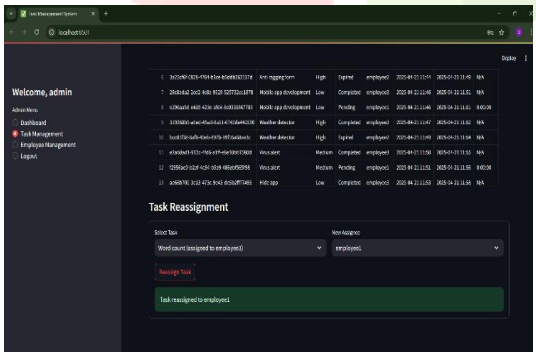



**Figure 5 Reassign**

The figure 5 shows the Task Reassignment Interface of the cloud-based Task Management System. It allows admins to manually or automatically reassign tasks if deadlines are missed. Task details and status are displayed in a table for easy management.

## VIII. CONCLUSION

The envisioned automated task rescheduling system has vast potential in simplifying task management in contemporary, distributed organizations. Through the integration of error-level detection, sequential reassignment algorithm, cloud database integration, and a friendly user interface, the system improves responsiveness, reduces project delays, and increases employee motivation through gamification. Its modular design facilitates future growth and adaptability, such as predictive analytics and integration with broader enterprise platforms. This approach is a solid basis for smart task management in the age of cloud computing and dynamic workplace.

## IX. REFERENCES

1. S. Singhal, N. Gupta, P. Berwal, Q. N. H. Naveed, A. Lasisi, and A. W. Wodajo, "Energy Efficient Resource Allocation in Cloud Environment Using Metaheuristic Algorithm," IEEE Access, vol. 10, pp. 10012-10024, 2022.

2. S. V. Aswin Kumer, N. Prabakaran, E. Mohan, B. Natarajan, G. Sambasivam, and V. B. Tyagi, "Enhancing Cloud Task Scheduling with a Robust Security Approach and Optimized Hybrid POA," IEEE Access, vol. 11, pp. 122426–122445, 2023.

3. J. Sun and H. Cho, "A Lightweight Optimal Scheduling Algorithm for Energy Efficient and Real-Time Cloud Services," IEEE Access, vol. 10, pp. 5697–5714, 2022.

4. N. K. Walia, N. Kaur, M. Alowaidi, K. S. Bhatia, S. Mishra, N. K. Sharma, S. K. Sharma, and H. Kaur, "An Energy-Efficient Hybrid Scheduling Algorithm for Task Scheduling in Cloud Computing Environments," IEEE Access, vol. 9, pp. 117325 117337, 2021.

5. A. Rahmani and H. Naderpour, "A Dynamic Task Scheduling Approach Using a Reinforcement Learning Framework," Journal of Cloud Computing, 2023.

6. R. Kumar and T. Kaur, "Cloud Task Scheduling Using a Multi-Objective Genetic Algorithm," Journal of Parallel and Distributed Computing, 2022.

7. M. Ashraf, K. Iqbal, and L. James, "Adaptive Task Scheduling in Cloud Computing with QoS Constraints," Future Generation Computer Systems, 2023.

8. H. Kumar and P. Singh, "Efficient Resource Allocation in Cloud Computing Using Machine Learning-Based Task Rescheduling," Cluster Computing, 2022.

9. Y. Zhang and Q. Li, "Task Scheduling Optimization in Cloud Computing Using Particle Swarm Optimization (PSO)," Journal of Cloud Computing, 2021.

10. S. Patel and T. Rao, "A Cost-Aware Scheduling Approach for Optimized Resource Utilization in Cloud," Journal of Supercomputing, 2022.