A

Tribhuvan University Affiliated College

Samriddhi College

A
Final Year Project Report
On

Task Management

(In partial fulfillment of the requirement for the Bachelor degree in Computer Science and Information Technology)

Submitted to:

Department of Computer Science and Information Technology,
Samriddhi College
Lokanthali, Bhaktapur

Submitted by:

Alisha Dhukuchhu(Roll no 11121/073)
Anjana Shyama (Roll no 11123/073)
Kanchan Nakarmi (Roll no 11132/073)
Rajeev Rajchal (Roll no 11147/073)

A
Tribhuvan University Affiliated College
Samriddhi College

A

Final Year Project Report
On

Task Management

(In partial fulfillment of the requirement for the Bachelor degree in Computer Science and Information Technology)

Submitted to:

Department of Computer Science and Information Technology,
Samriddhi College
Lokanthali, Bhaktapur

Submitted by:

Alisha Dhukuchhu(Roll no 11121/073)
Anjana Shyama (Roll no 11123/073)
Kanchan Nakarmi (Roll no 11132/073)
Rajeev Rajchal (Roll no 11147/073)

DECLARATION

Task Management Software which is being submitted to the Department of Computer Science and Information Technology, Samriddhi College, Lokanthali, Bhaktapur, Nepal for the fulfillment of the seventh semester as major project under the supervision of Assitant Professor Loknath Regmi.

This project is original and has not been submitted earlier in part or full in this or any other form to any university, here or elsewhere, for award of any degree.

…….	…….
Alisha Dhukchhu	**Anjana Shyama**
Roll no 11121/073	Roll no 11123/073
…….	…….
Kanchan Nakarmi	**Rajeev Rajchal**
Roll no 11132/073	Roll no 11147/073

Samriddhi College
[T.U. Affiliated]
Lokanthali-16, Bhaktapur

SUPERVISOR'S RECOMMENDATION

I hereby recommend that this project be prepared under my supervision by ALISHA DHUKUCHHU, ANAJANA SHYAMA, KANCHAN NAKARMI, and RAJEEV RAJCHAL entitled **TASK MANAGEMENT** in partial fulfillment of the requirements for the degree of B.Sc.CSIT Computer Science and Information Technology be processed for evaluation.

. . . . . .

Loknath Regmi

Virtual Lecturer

Samriddhi College

Lokanthali, Bhaktapur

Samriddhi College
[T.U. Affiliated]
Lokanthali-16, Bhaktapur

CERTIFICATE OF APPROVAL

This is to certify that this project was prepared by Alisha Dhukuchhu (Roll no 11121/073), Anjana Shyama (Roll no 11123/073), Kanchan Nakarmi (Roll no 11132/073) & Rajeev Rajchal (Roll no 11147/073) entitled **Task Management** in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology has been well studied. In our opinion, it is satisfactory in the scope and quality as a project for the required degree.

…….	……...	…….
Loknath Regmi	**(External Examiner)**	**Mr. Sandeep Shrestha**
Virtural Lecturer	CSIT	Principal
Supervisor	Tribhuvan University	Samriddhi College
Samriddhi College	Kirtipur, Nepal	Lokanthali, Bhaktapur
Lokanthali, Bhaktapur		

ACKNOWLEDGEMENT

With great pleasure, we present our project on the application “TASK MANAGEMENT”, whose major objective is to act as task planning management using criteria as deadline and profit.

We would like to thank Mr. Loknath Regmi, our Project Supervisor, who with his continuous support and effort, made the research upon the problem of task distribution. Similarly, we cannot stay without expressing our gratitude to Mr. Sandeep Shrestha, the Principal of Samriddhi College, for providing us an opportunity to perform our research activities and come up with a working solution that would be beneficial for a large mass of people.

We would also like to thank Mr. Bikash Balami and Mr. Arpan Kumar Maskey for their encouragement and guidance toward making this report as per the standard.

We are thankful and fortunate enough to get constant support from our colleagues and teaching staff of B.Sc.CSIT department, which helped us, complete our project. We would also like to extend our regards to all the non-teaching staff of B.Sc.CSIT department for their timely support.

Lastly, we are thankful to all the helping hands who always welcomed us with their warm support at the time when we needed them. Also, we would like to apologize for the mistakes that may have been committed in this project proposal.

Alisha Dhukuchhu(Roll no 11121/073)
Anjana Shyama (Roll no 11123/073)
Kanchan Nakarmi (Roll no 11132/073)
Rajeev Rajchal (Roll no 11147/073)

ABSTRACT

Task management is the process of managing a complex set of activities as per its deadline and profit performed by a variety of people that requires proper skills, knowledge, tools, and techniques to make task performance smooth and fast. Random division of tasks with random profit value affects the running of the project which sometimes leads to various difficulties along with a delay in task completion. For such reasons, we bought the task management system where complex sets of tasks are analyzed as per their deadline and profit which is assigned to related people accordingly. In this system, a set of the task is compared with each other as per its profit value, deadline and weight. Higher the weight of the task, the higher the priority to the task to be done first. As per the weight of the task, scheduling is done and distributed to the related personnel to perform the task.

This system once gets task sorted by simple admin, it starts to find relative attributes like profit value and deadline of the sorted task and task are firstly sorted as per its profit value. Then as per the Elo algorithm, the system further calculates the weight of the task based on its relative attribute i.e. profit and deadline. Such calculated weights are compared with each other to find a higher weighted task which is sorted in the first place and the remaining task in descending order. The goal is to break the task as per its deadline and allocate those tasks regarding its profit value that greatly influence the result of a project or task performance.

Keyword: *ELO, Greedy Job Sequencing Algorithm, Tasks, Sequence, Profit, Deadline.*

LIST OF ABBREVIATION

API	Application Programming Interface
CSS	Cascading Style Sheets
EA	ELO Rating Algorithm
GDA	Greedy Deadline Algorithm
HTML	Hyper Text Markup Language
JS	JavaScript

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF TABLES

CHAPTER 1 INTRODUCTION

1.1 BACKGROUND

Completing work on time helps to improve the quality of work and many others. In this busy environment, everyone wants their work to be done on time for their smooth work with high profit in their working system. No system has been designed and implemented for automatic classification of the project as per its deadline and profit value and tasks are carried out with random time allotment without determining deadline and profit value. Due to such unmanaged or random allotment of the time to any project without knowing its profit value and deadline directly affect profit value and timely completion. For such reasons, we have proposed a system for classifying the project as per its deadline allotted by comparing its profit value assigned to the task or project.

"Task management" is a system where we can determine the deadline of the task which is assigned by the client and profit value of different projects as it is also determined by the client and resource person. Comparison between the different tasks regarding its profit value is made and deadlines are first assigned to the higher profitable task as company work for earning profit where it helps in task management as per profit value and deadline of the project. Random management of tasks without any reference leads to a mess where important and profitable tasks can be missed out. During task management and scheduling, for projects with the same profit value and different deadlines we schedule the task of the lower deadline, and for projects with different profit values and same deadline, we schedule the task of higher profit value. If there is a single task then it is automatically scheduled to the system. Here onward, the proposed system will be able to help to increase the working efficiency with an effective and smooth outcome on the expected time. Division of task helps to complete the task on time where it gets enough time to check whether it works as per its aspects or not and also can be checked whether there is any dispute. If something like this is found then it can have enough time for its correction and testing

1.2 PROBLEM

Unassigned deadline for a task leads to wastage of time and delay in performing the task. Delaying in the task performance leads to a decrease in profit value. Unmanaged scheduling without any reference directly affects the smooth performance of the task on time. Due to a lack of determination of task completion time, tasks with a short deadline with more profit value are being missed and higher or more deadlined tasks with lower profit value are prioritized which directly affect in adding profit value. Therefore, it is most for determining the deadline of a task.

1.3 OBJECTIVES

The proposed system has the following objectives:

- To schedule the task as per its deadline.
- To determine the sequence of tasks performance deadline regarding its profit value.

1.4 SCOPE

The development of "**Task Management**" is primarily concerned with the profitability of the task completions with its scheduled time. The major scopes of the proposed project include the following

- Time management:
  The provided task can be completed on time as per the profit where other pending tasks could be done on time within its deadline.
- Task organization: Despite verifying the nature of the task, it can be organized systematically with an assigned deadline which helps to perform the task smoothly on time by a resource person

1.5 LIMITATIONS

The limitations of the system are given below:

- The deadline of a given project must be known.
- Lower profit values with higher deadline tasks is neglected.

CHAPTER 2 LITERATURE REVIEW

A system is being developed by Richard Eisenstein to classify tasks based on the actual weight and rating calculated on the basis of their deadlines and profit. After performing research, the team decided to model team players and predict the outcome of games based on "The ELO Rating System" [1]. Arpad Elo developed the Elo Algorithm for chess tournaments, basketball games, football games, and even for rating players in multiplayer video games. Using an Elo-based model based on a player's plus-minus score, the paper examined an individual's basketball strength. By focusing primarily on points, plus-minus was not able to accurately measure the contribution of a player, such as good defence, setting up screens, or sledging the other team. In the next step, an average rating was generated for each team, along with a pairwise comparison i.e. "Multiplayer ELO Rating System" [2] of each team's rating to calculate the probability of victory by them. Unlike the Elo rating system, this method offered more information than just predicting wins and losses since individual ratings were assigned to each player rather than just a team's rating.

$$R_{inew} = R_{iold} + K(s_{ij} - \pi_{ij})$$

Equation 2-1 Finding Relative Weight (R_{inew})

The K parameter controls how much weight should be given to a recently concluded game as compared to past games (prior information).

The value of Sij is used in the equation for updating Elo. The score considered is the win/loss information. Here is a definition for Sij:

$$S_{ij} = \begin{cases} 1, if\ Team_i\ beats\ Team_j \\ 0, if\ Team_j\ beats\ Team_i \\ 0.5, if\ Team_i\ draws\ Team_j \end{cases}$$

Equation 2-2 Score (S_{ij})

Parameter π_{ij} measures the expected win/loss of the Team against the Team.

Based upon that system, the winner or loser of the game was determined or predicted. Here in our project, we applied that approach to calculate the task rating based on the task's profit

and deadlines. As a result of comparing the calculated ratings of the tasks, the highest scoring task was given first place, followed by the rest in chronological order.

According to Louis L. Thurstone, “A Law of Comparative Judgment” [3] has been formulated and expressed as an equation. Psychological scales and continuums were defined by this law. The compared stimuli were then allocated along the continuum. It had expressed the experimentally observed proportion, p1>2 the judgments 1 (better, lighter, more excellent) performed better than 2 ' based on the values of the scales, the discriminably spreading of the different stimuli, and the correlation between their rejections. Experimentally and objectively, the validity of the psychological continuum definition was examined. There could only be consistency for one distribution of discriminably processes as a basis to define the scale.

Standard error of the percentage of judgments observed the observation equations for each of the five cases considered should each be weighed by a factor shown in the equation which will apply to the observation equations in any of the five cases. Application of the formula to the equation provides the weighted observation formula. In this paper, it was said that the judgment of error observed was not uniform, and so a factor should be considered that would be uniform for everyone, which is the k factor used in ELO rating.

In 1805 and 1808, Gauss, Legendre, and Adrain independently developed “The Least Square Method” [4]. In general, LSMs were used in data fitting, with the best fit minimizing residual squared sums. The Least Square Method was used to forecast sales for a company in that paper. By using the LS method, the study aimed to predict an organization's future sales for the year. Due to lack of forecast data in the past, several businesses had collapsed, so by using the Least Square Method, companies could forecast their future sales using the previously obtained data for a proper plan that could lead them to fulfilling their set goals and targets. Our team used this approach to determine the rating for each task based on their previous rating, using historical data as reference.

Therefore, the system “Task Management” has been developed with the reference of an Elo Rating where we applied that approach to calculate the task rating based on the task's

profit and deadlines. As a result of comparing the calculated ratings of the tasks, the highest scoring task was given first place, followed by the rest in chronological order. A law of comparative judgment has been formulated and expressed for the judgment of error observed and so a factor should be considered that would be uniform for everyone, which is the k factor used in ELO rating. the Least Square Method algorithm was applied to determine the rating for each task based on their previous rating, using historical data as reference.

CHAPTER 3 SYSTEM ANALYSIS

3.1 REQUIREMENT ANALYSIS

3.1.1 FUNCTIONAL REQUIREMENT

The functional requirements of the proposed system are listed below:

i) Job/Task List Input: Initial step to feed the job/task needs to be processed. The data system gets input.

ii) Breaking Down: Job/Task can be input on bulks so, breaking into individual tasks with their respective profit as well as deadline.

iii) Comparison: It performs the core operation of the system by comparing the deadline and profit with each task.

3.2 USE CASE DIAGRAM

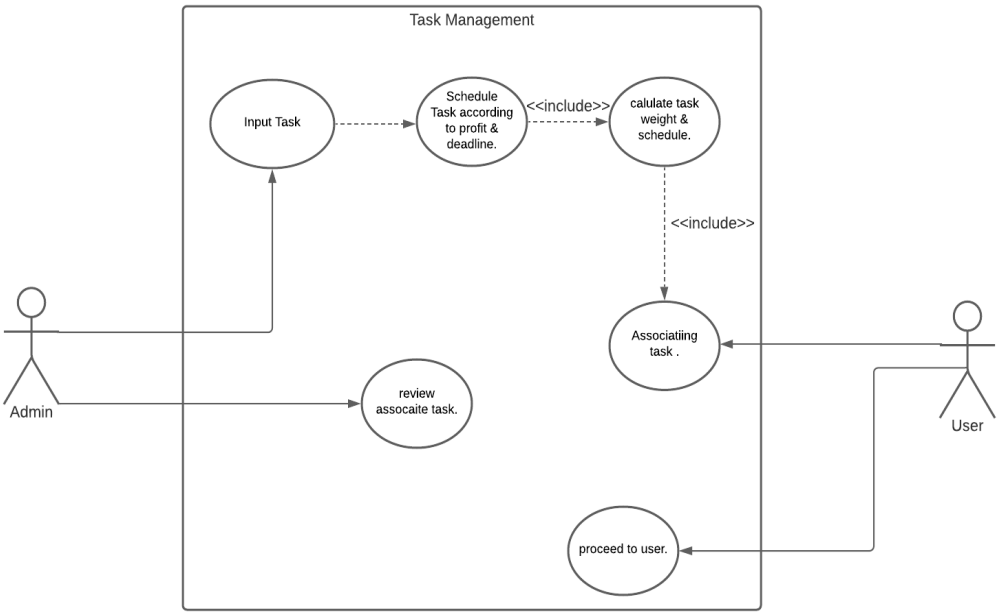


Figure 3:1 Use case diagram of task management.

In this system, admin provide the task. Inserted task is sent to the application to analyze the task's as per the greedy job sequencing algorithm. GJSA analyze the profit and deadline of the tasks then find the task with shortest deadline with higher profit value. Then schedule the task from higher profit value with lower deadline to the lowest profit value to higher deadline from top to down in chronological order. Some unscheduled tasks are sent pack to the application to reschedule with other input task and scheduled task are sent to the ELO Rating Algorithm. Where this system rates the task as per relative weight and k factor.

3.2.1 NON FUNCTIONAL REQUIREMENTS

The non-functional requirements of the proposed system are listed below:

i. Reliability: This system is reliable as it tends to give accurate results. The reliable output is produced with the help of this system because different giving reliable outputs will be used.
ii. Timing Constraints: The proposed system will take a finite amount of time and the processing takes a minimum amount of time to provide the result.

3.3 FEASIBILITY STUDY

3.3.1 ECONOMICAL FEASIBILITY

Through the use of open source technology in its development, the resource for the development process of the proposed system is minimal but the effort is above the normal standard. The frameworks, modules, and IDE that will be used are available for free on the web. This system opens a wide range of areas and can replace much other software through its further implementation.

3.3.2 TECHNICAL FEASIBILITY

The programming language and tools that we used are HTML, Javascript, Job sequencing algorithm, Elo algorithm.

3.3.3 OPERATIONAL FEASIBILITY

With the help of this system, it makes a comparison of the deadline of the tasks regarding its profit value to help to complete the task on a given deadline by the respective resources person. It makes the work efficient and effective. This system is user-friendly.

CHAPTER 4 SYSTEM DESGIN

4.1 FLOW CHART

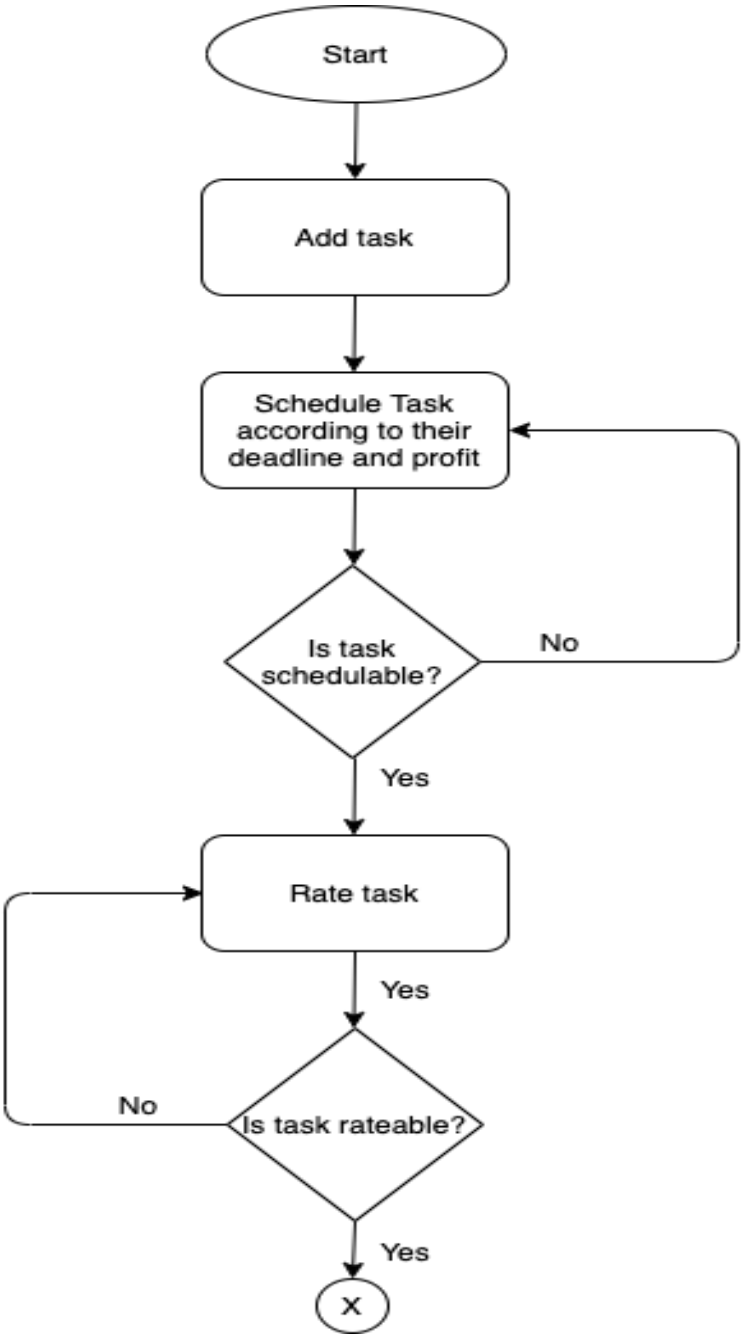


Figure 4:1 Flow chart

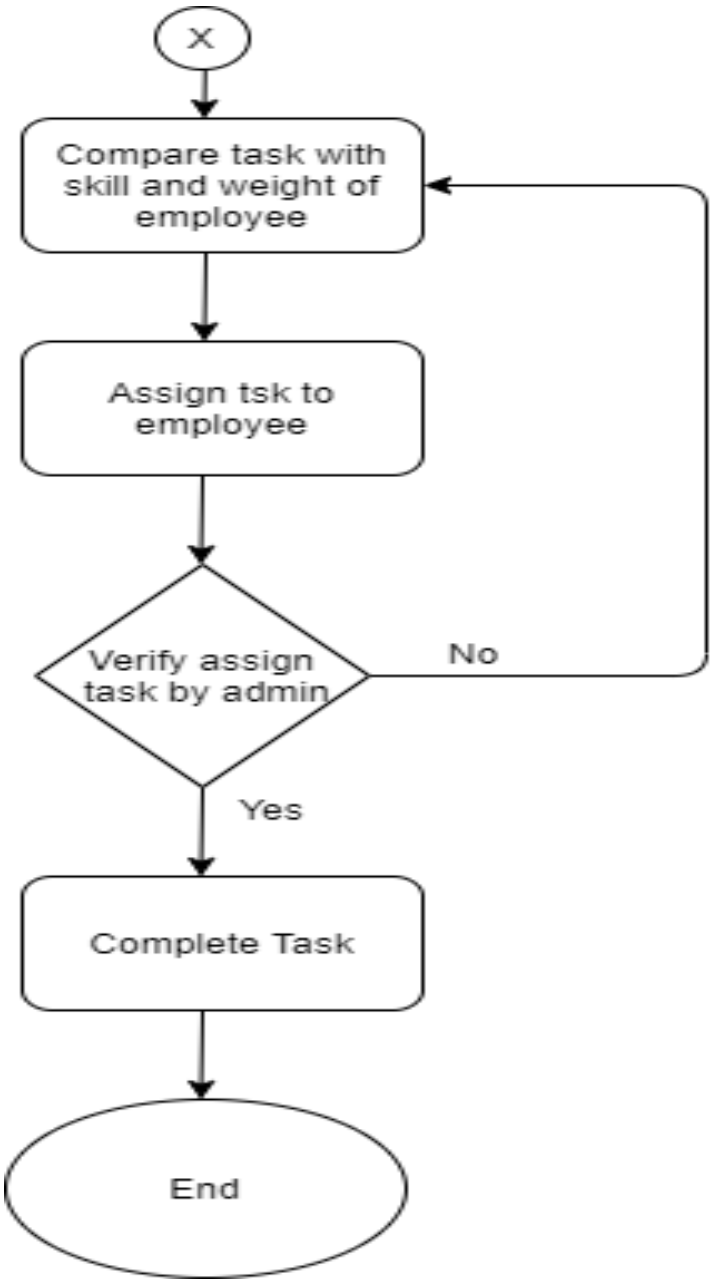


Figure 4:2 Flow Chart

In this system, admin will add the available tasks.In the system we have used the greedy job sequencing algorithm which will schedule the task according to their profit and deadline.The system will analyze the profit and deadline of the tasks and find the task with shortest deadline with higher profit value.Then those task which are schedulable will be scheduled according to the higher profit value and lower deadline to lower profit value with higher deadline.If the task are not schedulable it will send bask to system to reschedule in another sprint. After the schedule is set, ElO algorithm will evaluate the schedule for rating. ELO algorithm will use the constant k factor and relative weight to rate the task. k factor is a constant which is used to measure how strongly it influence the rating of task.
If K is of a lower value, then the rating is changed by a small fraction but if K is of a higher value, then the changes in therating are significant.In this system we choose the k factor via dviding 7 days of working hours i.e. 168 hours. Relative weight is calculated according to the previous weight of the task but, since we don't have previous information about task we will set it to 1.While comparing two task the winning task will have score 1 ,losing task will get 0 and if there is draw both will get 0.5.ELO algorithm is all based on probability. The expected probability is calculated from k factor, relative weight and score.Then the algorithm will rate the task from higher ELO rating to lower ELO rating

The unratable task will be send back to the system to rerate.When the rating is done, the task is compared with the skill and weight of employee. It will be done via inRange() function.The inRange() function will compare with the upperbound array and lower bound array of the employee's skill and weight.If the inRange() function matches between the range of employee's skill and weight range,he task will be assigned to the employee.The assigned task will be verified by the admin.Just in case the assigned employee is not approved, the task has to be again compared with the weight and skill of another employee. The verified assigned task will be completed by employee.

4.2 SEQUENCE DAIGRAM

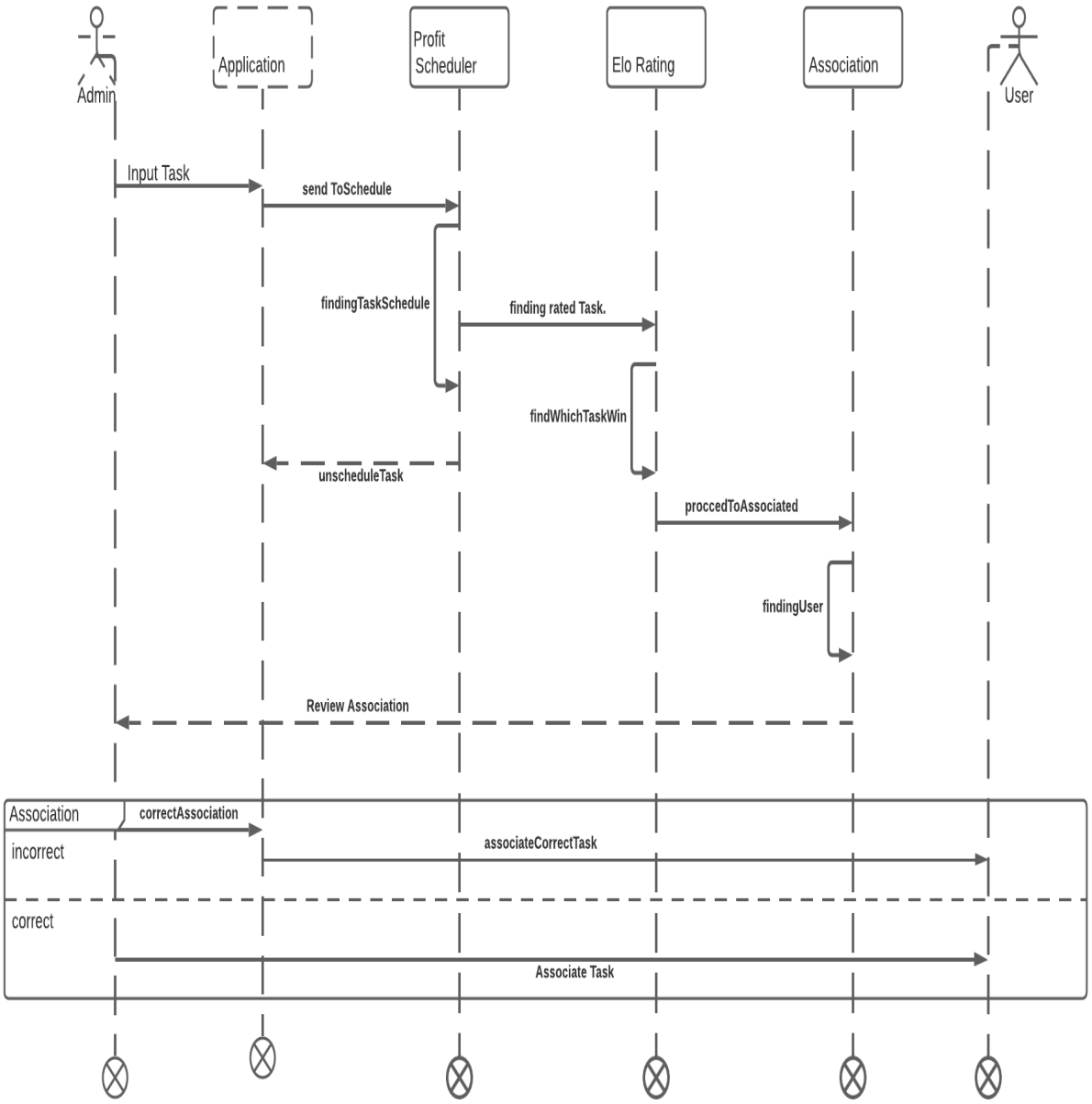


Figure 4:3 Sequence Diagram

In this system, admin provide the task. Inserted task is sent to the application to analyze the task's as per the greedy job sequencing algorithm. GJSA analyze the profit and deadline of the tasks then find the task with shortest deadline with higher profit value. Then schedule the task from higher profit value with lower deadline to the lowest profit value to higher deadline from top to down in chronological order. Some unscheduled tasks are sent pack to the application to reschedule with other input task and scheduled task are sent to the ELO Rating Algorithm. Where this system rates the task as per relative weight and k factor.

CHAPTER 5 IMPLEMENTATION AND TESTING

5.1 IMPLEMENTATION

Task management system helps to manage the task to take the workflow of the task. This system possesses to apply an ELO Algorithm and Greedy Job Sequencing algorithm to produce the workflow of the two done on sequential order.

1. Analysis and Design tools

   Various way of designing tools to create figures and diagrams like entity relationship diagram flowchart, use-case diagram.

2. Tool Implementations

   In this system/design of the system we have use Nodejs a framework of JavaScript for implementation of the project.For simple backend we use express and for simple frontend was build using react JS. Overall, the single bunch of code is implantation on next JS.

5.2 METHODOLOGY

In this proposed system, the system is operated by normal person for find schedule workflow of the task. The process of this system is break into three functions

i. Input of task list.
ii. Finding the task schedule based on the profit.
iii. Calculating the task weight with respect to time, deadline & its relative weight

5.2.1 GETTING INPUT:

Input format are most mandatory part of the system. For this system, the input data are collected with the json or object format on the JavaScript.

The format of json object might be in array with attribute of:

i. relative_weight
ii. job
iii. win_count
iv. deadline

v. profit

5.2.2 PROCESSING DATA

This system process data in two different phases. First the given data are schedule or order based on the deadline & profit. Then applied Greedy Job Sequencing algorithm for finding the job schedule with the appropriate task. Then the schedule moves to the second phase i.e., Elo algorithm which find the relative weight using its relative weight, kfactor etc. Then the task moves to find its relative uses with inrange js method to find the appropriate user.

5.2.3 ALGORITHM

a. Greedy Job Sequencing Algorithm

It provides sequence of job based on their profit and deadline being greedy over those tasks. It is a simple algorithm.

```
for i = 1 to n do
        Set k = min(dmax, DEADLINE(i)) //where DEADLINE(i)
        denotes deadline of ith job
          while k >= 1 do
            if timeslot[k] is EMPTY then
              timeslot[k] = job(i)
              break
              Endif
                  Set k = k - 1
          Endwhile
endfor
```

b. ELO Rating algorithm

ELO is used to determine the output of a game by using player's rating. Elo is based on the probability of wining. "Player with a higher rating have higher probability of winning a game than players with lower rating. In this proposed system, Pairwise Elo rating algorithm was used for finding the task rating.

Here the k factor is the strong match measure that influences a player rating. If k is of a lower value the rating is changed by a small fraction but if k is of higher value, then the changes in the rating are significant.

In this system, we choose the k factor via dividing 7 days of working hours i.e.

7*24=168 hours

So, we divided the 168 hours by 48.

48 is the constant value to find the k

i.e. $k = \frac{168}{48} = 3.5$

Since k doesn't have universal value

a. The subscripts 'a' and 'b' to denote the player. Notice: this equation is for computing expected (E) probability for player A and the difference is Rb — Ra. There is a similar converse equation for player B with the subscripts reversed so that it contains

$$E_b = \frac{1}{1 + 10^{\frac{(R_a - R_b)}{400}}}$$

Equation 5-1 Finding Excepted (E_b)

$$R`_a = R_a + k(S_a - E_a)$$

Equation 5-2 Finding New (R_a)

b. The usage of the constant 400 can be adjusted based on how you want the expectations to correlate to the differences in ratings and we'll look at this it in detail more later in the article.

a. The K for K-factor and is another constant that can be e adjusted to control the sensitivity of the of the update. Too large and it will be to sensitive. Too little and it won't be responsive.

b. The S is from the word Score which is a fixed value in the range of the probability. 1 for a win, 0 for loss, and optionally 0.5 for draw if your game to supports this. I often refer to score as outcome in the article.

5.3 TESTING

5.3.1 UNIT TESTING

Unit testing is the level of software testing of the project's system in which the smallest testable parts of a system called unit is individually tested. Unit testing concentrates on each unit of the system as implemented in the source code. The main purpose of unit testing is to validate each unit of the software to perform as designed.

S.N.	Test Case ID	Test Scenario	Test Input Data	Test Steps	Expected Results	Output	Remarks
1	TM-001	Running App On Web	n/a	Start server using command (yarn start)	App must run of defined port on web interface.	App ran successfully	Successful initialization
2	TM - 002	Data Input to GJSA	List of task data in json format	Feed on API body.	Task must be schedule according to profit and deadline	Task has been scheduled with its profit and deadline.	Successful task scheduled.

Table 5-1 Test Case App State

S.N.	Test Case ID	Test Scenario	Test Input Data	Test Steps	Expected Results	Output	Remarks
1	TM-E001	Finding relative_weight and win_count of given task.	Task sequence provide by GJSA	Automatic task sequence input by GJSA code.	Must return relative_weight and win_count.	Successful return of the job sequence with relative wight and win count.	Pass
2	TM-U001	Assign User	Input user data and task schedule from elo	Input task schedule from elo	Must assign task to user and return the task lsit.	User has been assigned.	Pass

Table 5-2 Test Case App Process

CHAPTER 6 CONCLUSION AND ENHANCEMENT

6.1 CONCLUSION

The system 'Task Management' has been constructed to make smooth and proper distribution of the task in any field. Systems like Jira and Trello distribute the task manually only. But this system distributes the task regarding the deadline and profit of the task. A comparison between the deadline and profit of different tasks is made to find the higher-profit task with a lower deadline which is scheduled first and remaining tasks are scheduled automatically with higher weight to lower.

6.2 FUTURE ENHANCEMENT

The possible future enhancement of the system are listed as below:

i. Weight calculation of tasks is done with greater efficiency and accuracy.
ii. The deadline of the task is assigned automatically as per the nature and weight of the task.
iii. For this project, we have used the "Pairwise Approach" of the Elo Rating Algorithm. So in the future, we can use a different method of the Elo Rating Algorithm so that there is fair play on the system.

REFERENCE

[1] A. Mishra, "Elo rating system: Common link between Facemash and chess!," 23 September 2016. [Online]. Available: https://www.hackerearth.com/blog/developers/elo-rating-system-common-link-facemash-chess/?fbclid=IwAR3m9Ik9_bbRPfNWpn_nJ9ZagwjPZWXgxXIxSN3kYf-SKEWG7aEgLghnIsY.

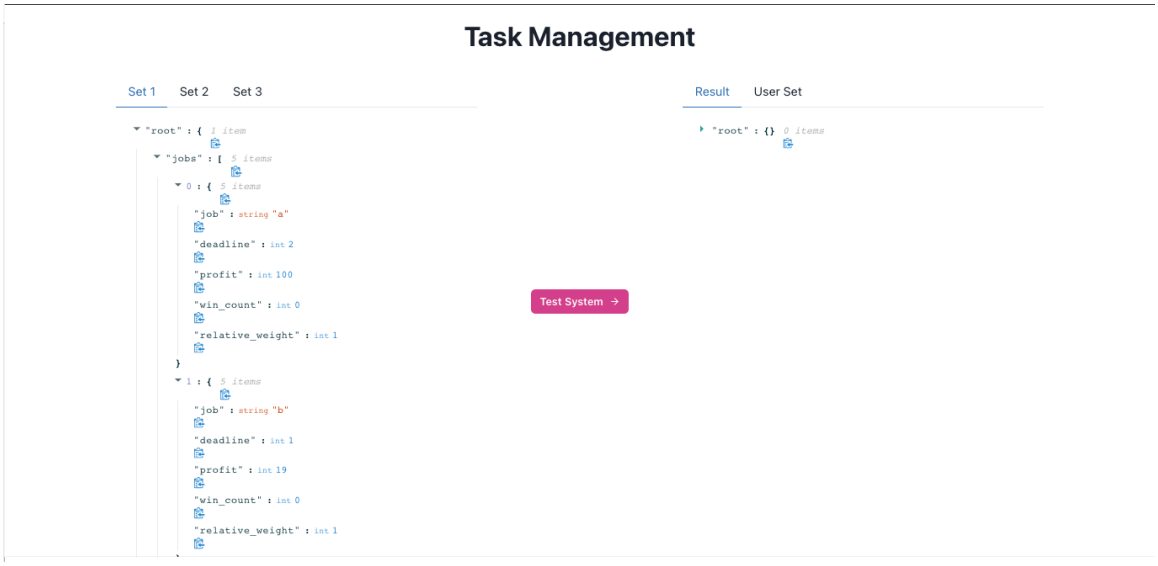
[2] G. Narula, "Building a Multiplayer Elo Rating System," 16 December 2012. [Online]. Available: https://www.gautamnarula.com/rating/.

[3] L. Thurstone, "A law of comparative judgment," *A law of comparative judgment,* vol. 34, no. 4, p. 273, 1927.

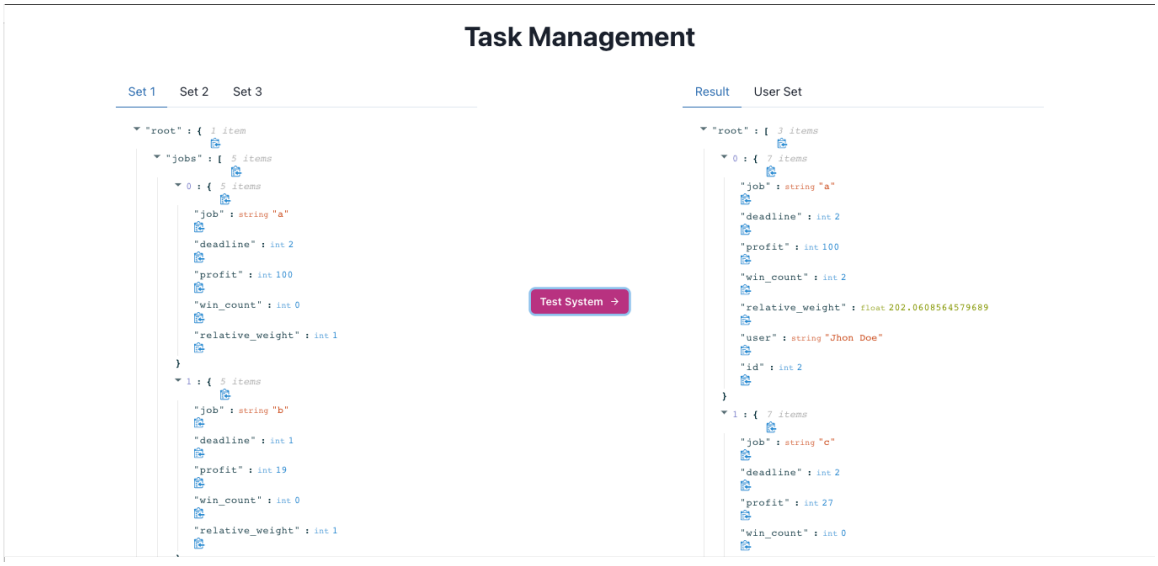
[4] GeeksForGeeks, "Job Sequencing Problem," 19 Jul 2021 . [Online]. Available: https://www.geeksforgeeks.org/job-sequencing-problem/.

APPENDICES

1. Screenshot of minimalistic UI.



2. Task Assigned (Result of processing.)



3. Greedy Job Sequencing Algorithm

```
jobSchedulingList: (sortedJobs) => {
        const scheduleJobs = []
        //length of the original jobs array.
        const sizeOfJobs = sortedJobs.length
        for (let i = 0; i < sizeOfJobs; i++) {
           for (let j = Math.min(sizeOfJobs, sortedJobs[i].deadline) - 1; j >= 0; j--) {
              if (!scheduleJobs[j]) {
                 scheduleJobs[j] = sortedJobs[i];
                  break;
              }
           }
        }
        return scheduleJobs
    },
```

4. ELO Rating Algorithm

```
eloRateTask: (tasks) => {
                /*
                * k is factor that
                * K = γt, where t is ti/48, ti is time and 48 is normalize constant.
                *     γ is choose by running synthetic pr real-life data represent by d
                *  we choose γ value as 1.
                * every sprint is of 7 days i.e. 168 hr
                */
        const k = 168 / 48 //generating custom k factor constant value
        const taskCopy = [...tasks]

        const comparison = (winner, looser) => {
          //expected rating calculation.
           const expectedRatingWinner = winner['relative_weight'] === 1 ?
          winner['deadline'] * winner['profit'] : winner['relative_weight']
          const expectedRatingLooser = looser['relative_weight'] === 1 ?
          looser['deadline'] * looser['profit'] : looser['relative_weight']
                //probability to win
           const probabilityLoser = 1.0 / (1 + Math.pow(10, (expectedRatingWinner
          - expectedRatingLooser) / 400))
          const probabilityWinner = 1.0 / (1 + Math.pow(10, (expectedRatingLooser
          - expectedRatingWinner) / 400))
                //comparison
          if (probabilityWinner > probabilityLoser) {
            winner["win_count"] = winner["win_count"] + 1
            looser["win_count"] = looser["win_count"] - 1
                winner["relative_weight"] = expectedRatingWinner + k * (1 -
                probabilityWinner)
            looser["relative_weight"] = expectedRatingLooser + k * (0 -
            probabilityLoser)
```

```
    } else if (probabilityWinner === probabilityLoser) {
      winner["win_count"] = winner["win_count"]
      looser["win_count"] = looser["win_count"]
    } else {
      looser["win_count"] = looser["win_count"] + 1
      winner["win_count"] = winner["win_count"] - 1
       winner["relative_weight"] = expectedRatingWinner + k * (0 -
    probabilityWinner)
      looser["relative_weight"] = expectedRatingLooser + k * (1 -
    probabilityLoser)
    }
  }

  for (let i = 0; i < taskCopy.length - 1; i++) {
    for (let j = i + 1; j < taskCopy.length; j++) {
      if(typeof i === undefined){
        i++
      }
      if(typeof j === undefined){
        j++
      }
      if(typeof i !== undefined || typeof j !== undefined){
        comparison(taskCopy[i], taskCopy[j])
      }
    }
  }

  return taskCopy.sort((itemA, itemB) => {return itemB.win_count -
  itemA.win_count});
}
```