**Problem Statement:**

Efficient task management for a team, or even for a person for himself, is one of the major challenges in software engineering. There are many platforms (e.g. trello, GitHub) which helps in this cause by allowing users to maintain lists of their “to do” tasks alongside with necessary information. But sometimes an engaging interaction may be needed for the user to really help him/her get going. For example, a person may write down his ‘to do’ tasks but in real can have no impetus to further checking it in due time. But if there were an entity like a real-life person who would constantly and efficiently track the progress and notify the user working as a smart reminder, more can be gained.

Also from a team point of view, often an independent entity is required to continuously evaluate each member’s performance and maintain a leaderboard to provide public rewards/penalties for the members to provide an impetus. Many artificial solutions are already there, but making them as interactive and engaging as possible remains a problem in the domain.

**Bot Description:**

Although efficiently performing these jobs directly contributes to the eventual performance gain, the uninteresting nature of them makes it difficult to attract people to perform them. Moreover, the repetitive and rule-based logics of such tasks make it possible for automation, thus saving a lot of resources in process. Thus, a bot could be the best solution to address jobs mentioned in the problem statement having a constant presence in those tasks managing platforms just like a real-life person.

Our bot doesn’t have conversation in a typical meaning with the user, but it would ask the users for different inputs from time to time, store them in its memory, and send reminders/suggestions/motivations in a message like manner. The best fit for the bot could be the “Space Responder” category.

**Use Cases:**

**[1] Send Nagging Reminder**

1. Precondition: User must give due date and expected hours for completion when adding a new task to the list. User optionally is expected to update the progress of the task also.
2. Main Flow: Bot will track the timeline of the task and progress [S1]. If it’s necessary, it will send reminders [S2]. It can continuously send reminders until the user takes an action [S3].
3. Sub flows:  
     
   S1: Bot would have a logic system to calculate checkpoints. (e.g. 2 days before the due date if the task has 6 hours of equivalent work remained incomplete)  
     
   S2: It will send notifications to the user. (e.g. send a mail to user’s id)  
     
   S3: It will wait for a certain amount of time for user’s response. In case of no response, it will keep sending reminders in a loop until the due date.
4. Alternative flow:   
     
   A1: If the user hasn’t added due date and other information for a task in the list, it will continue asking those information (via mail) at certain intervals.

**[2] Calculate Rewards based on Team Members’ Performance**

1. Precondition: With addition to the preconditions for use case 1 where user add tasks with due date and predicted no. of hours to complete it and optionally updating progress,

User must also input the completion date and no. of hours after finishing the task.

1. Main Flow: At certain intervals[S1], the bot will evaluate each member's’ performance based on a rule-based logic system and calculate score & rewards[S2], and update the leaderboard of the team[S3] and [S4] will give new targets and set new competitions for the members

(e.g. the rule-based logic system may have a logic to give more score the faster one completes the task. Detail logic will be added later)

1. Sub flow:  
   S1: It will keep track of time and will get activated after certain intervals.  
   S2: It will evaluate all members’ performance based on current information available according to its logic system and calculate score & rewards   
   S3: It will update the leaderboard for the whole team

S4: It will set new targets for each member and set up new competition (weekly?)

1. Alternate Flow:  
   A1: In absence of adequate information for any user, it will add penalties(minimal) for inactivity to his/her score in the leaderboard.

**[3] Reminder Buddy**

1. Precondition: same as use case 1 & 2.
2. Main flow: The bot will ask for progress to each member based on his tasks and due dates [S1], it will wait a certain period (differing on scenario, periods will be dynamically calculated) [S2], if response is positive, it will post congratulatory message on a public channel and award score in the leaderboard[S3], if response is negative, it will post reminder message to a public channel and add penalties in his score in the leaderboard
3. Sub flow:  
   S1: The bot will ask for progress to each member based on his tasks and due dates  
   S2: it will wait a certain period (differing on scenario, periods will be dynamically calculated)  
   S3: if response is positive (response of progress), it will post congratulatory message on a public channel and award score in the leaderboard  
   S4: if response is negative (either no response/response of no progress), it will post reminder message to a public channel and add penalties in his score in the leaderboard
4. Alternate Flow:  
   A1: If information is missing about any member, it will post message to public channel that the member is inactive