**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 intelligently assign new tasks to the right person based on his/her skillset and current workload. Many artificial solutions are already there, but making them as interactive and engaging as possible still 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 task 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 Smart Reminder**

1. Precondition : User must give due date and optionally 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. Subflows:  
     
   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] Evaluate Team members’ Performance**

1. Precondition: Users must add their task in the list with its due date, completion date, predicted and required no. of hours for completion of the task in the platform.
2. Main Flow: At certain intervals[S1], the bot will evaluate each member's’ performance based on a rule-based logic system[S2], and send appropriate messages from a list hard coded in its memory.
3. Subflow:  
   S1: It will keep track of time and will get activated after certain intervals.  
   S2: It will evaluated all members’ performance based on current information available according to its logic system and select appropriate message for all of them.  
   S3: It will send (via mail) them to the members.
4. Alternate Flow:  
   A1: In absence of adequate information, it will select generic motivating messages for the users.

**[3] Intelligently Assign Tasks**

1. Precondition: All members will always have their updated skillsets in the platform.   
    When adding a task, the manager must give input the required skillsets for the task.
2. Main flow: The bot will read skillsets[S1] and current workloads of the members[S2] from the platform and its own memory, match them based on a rule-based logic system[S3] and assign the task to one/more members.[S4]
3. Subflow:  
   S1: The bot reads skillsets from the platform.  
   S2: The bot predict free work hours available for each member based on the current information.  
   S3: Run these data into a simple logic based function and outputs results assigning one/more members to the task specifying the no. of hours assigned to them.  
   S4: Send them mail notifying the assignment. (sending them repeatedly at certain periods unless a response is received)
4. Alternate Flow:  
   A1: If too many information is missing, It doesn’t produce results, and notify project manager to handle the job himself. (deflecting duty)