



# P-11: REMINDER AND TASK SCHEDULER

Team: ASAN

- NEELABH RANA(202301476)
  - SOHAM MEVADA (202301484)
  - ATIK VOHRA (202301447)
  - AKSHAT BHATT (202301460)
- 

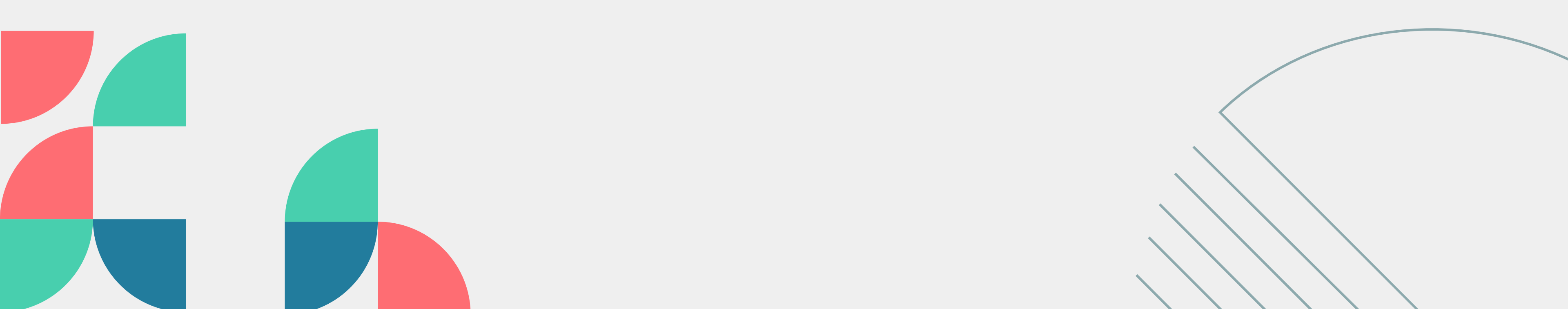


**"STAY ON TOP  
OF YOUR TASK WITH  
EASE"**



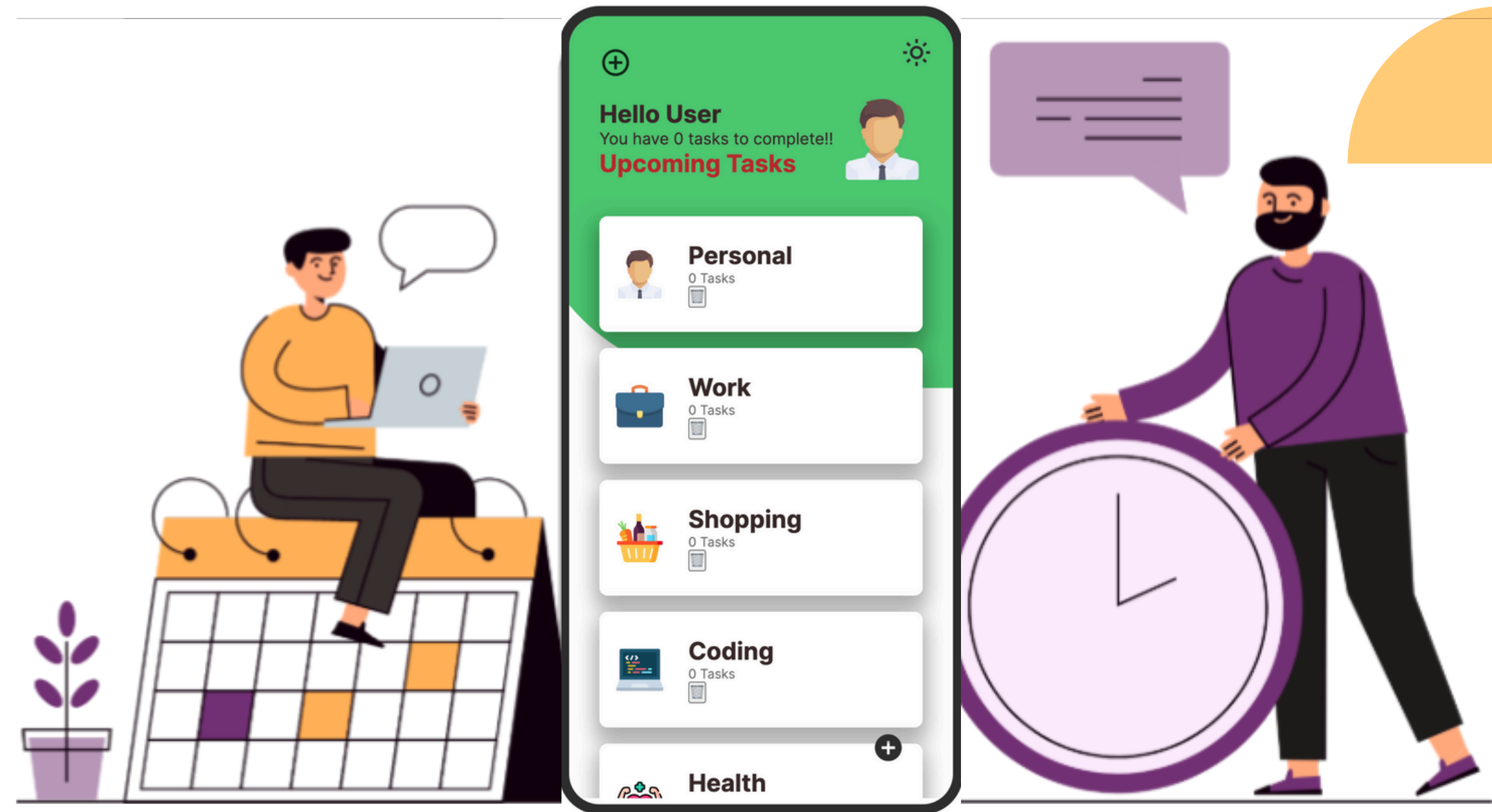
## P-11. REMINDER AND TASK MANAGER

Create an application that manages tasks and schedules using data structures to prioritize and track tasks based on deadlines or importance levels. It should also have reminder functionality as the deadline is approaching.



# Solution:

We have made console based app(using CPP) and a Graphical User Interface (using HTML,CSS,JS).To solve the core problem of task i.e. rescheduling and reminder we have used Data Structures and used some algorithms to come up with the final solution.



Now lets dive into the Algorithms ....

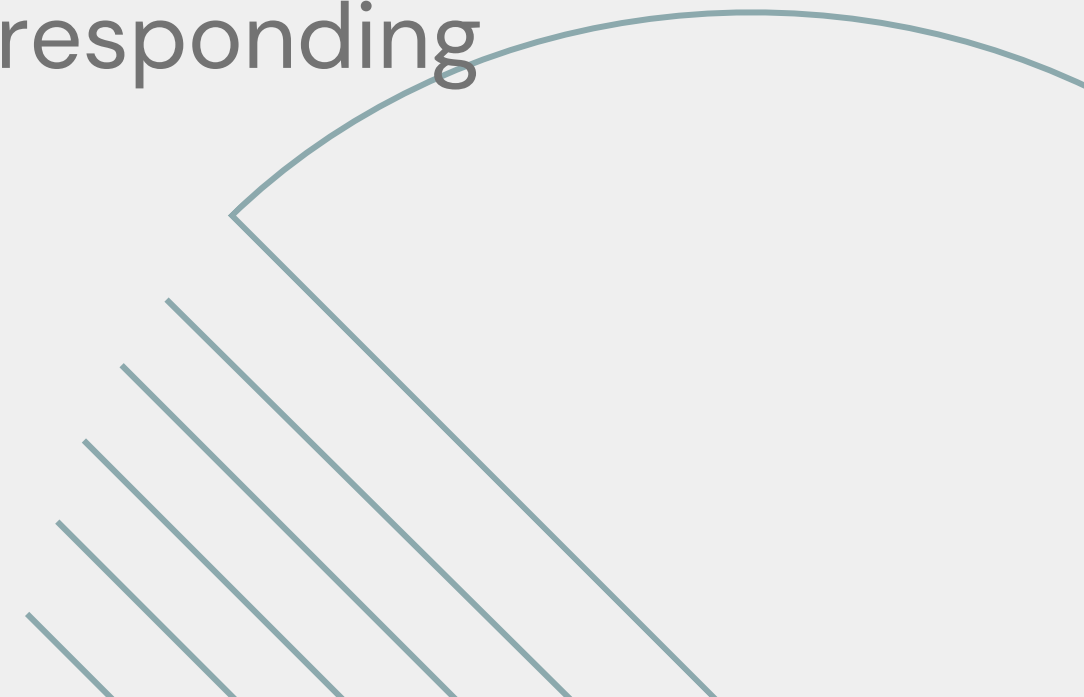
# ALGORITHM



We have used Doubly Linkes List for arrangement of tasks in systematic and required way...

Once the Program is terminated the data given by the user is not erased but it is rather stored in file using the concept of file writing.

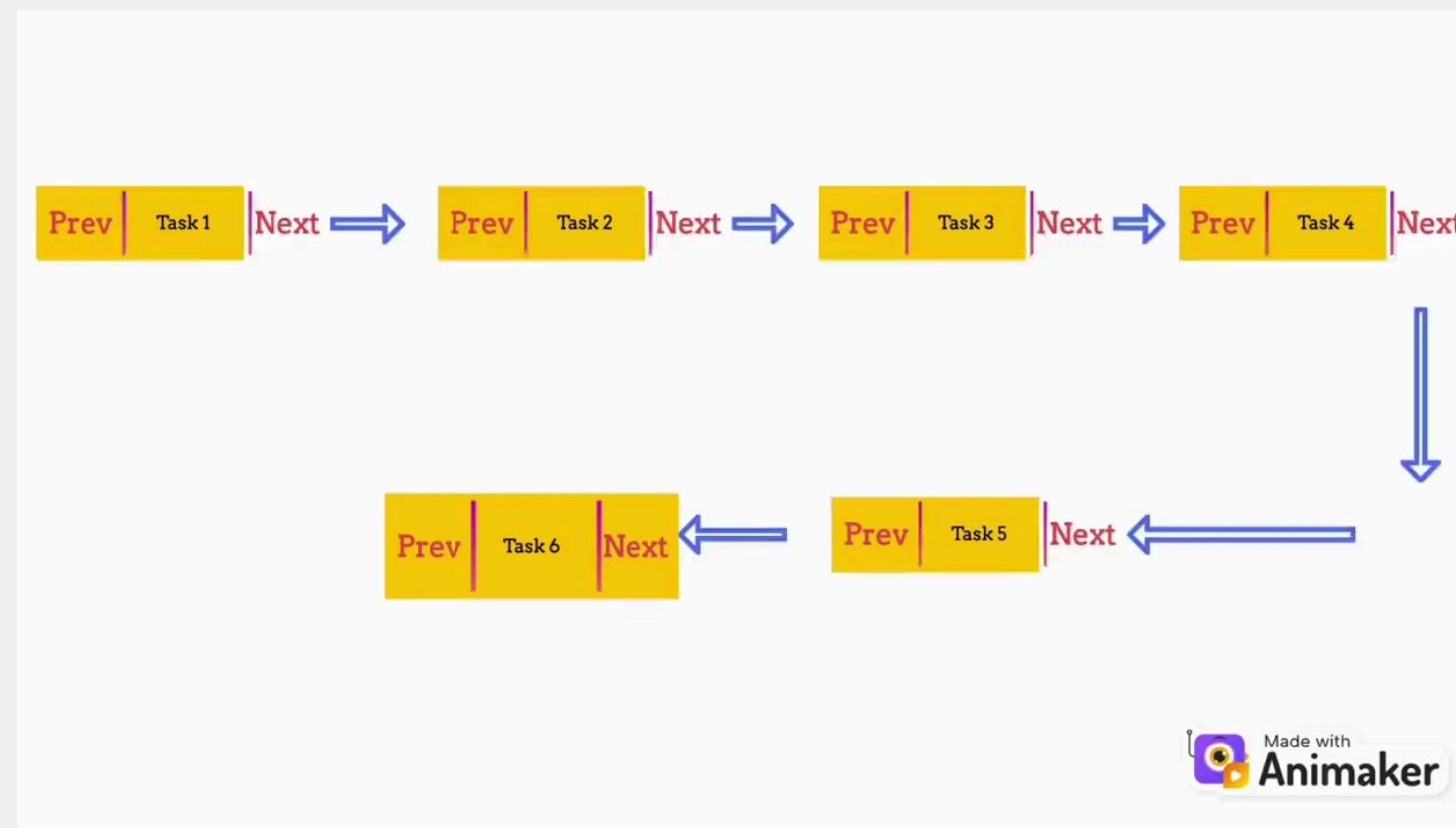
The content of the files are rewritten into the corresponding linked list every time the code is run.



# ALGORITHM

The program is made completely dependent on user responses. For the insertion of task, valid input data is taken and then a Node is made which is then inserted in the list as shown in the clip below.

Time complexity of insertion:  $O(n)$   
Space complexity of insertion:  $O(1)$   
(where  $n$ =number of nodes in the list.)

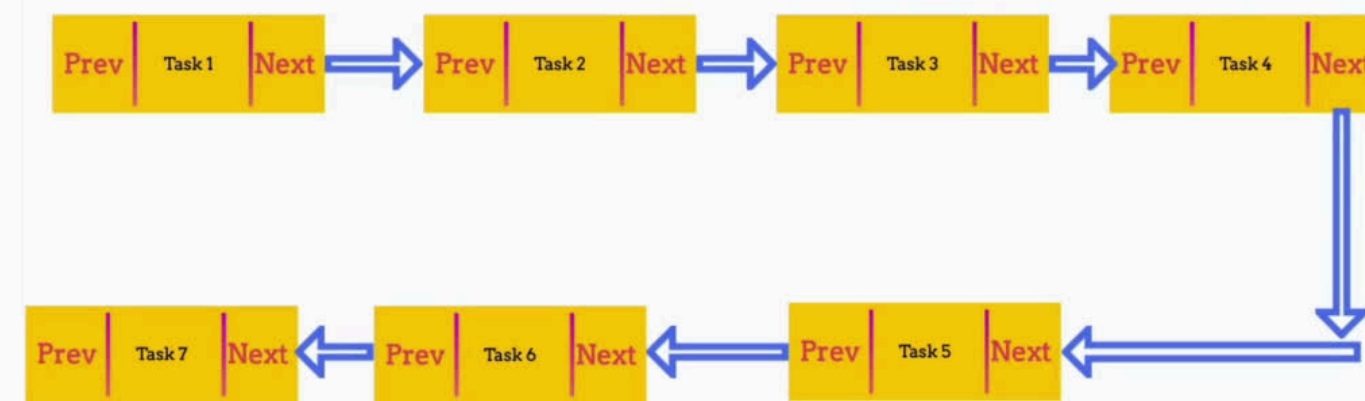


# ALGORITHM

If we want to delete a particular task then we will search through the list and find the task and make required changes for the deletion of the task(Node).

Time complexity of deletion:  $O(n)$

Space complexity of deletion:  $O(1)$   
(where  $n$ =number of nodes in the list.)



First we will find the task to be deleted and then delete it and change the links accordingly


Made with  
**Animaker**



# ALGORITHM

We have made a reminder function which scans the task within 12hrs and reminds the tasks coming within that time period.

The time period reminder i.e. 12 hr. can be changed just by changing the value given in the function call statement of reminder function in the main function.





# ALTERNATE ALGORITHM

We have made another implementation which uses hash tables instead of linked lists, the main purpose for this is to change the time complexity of insertion and deletion from  $O(n)$  to  $O(1)$ .

There is a function which calculates a specific value or key for each entry of date, time, priority which is also known hash function. According to the hash value, the file is stored inside the system's memory and which has the address of the hash value created.

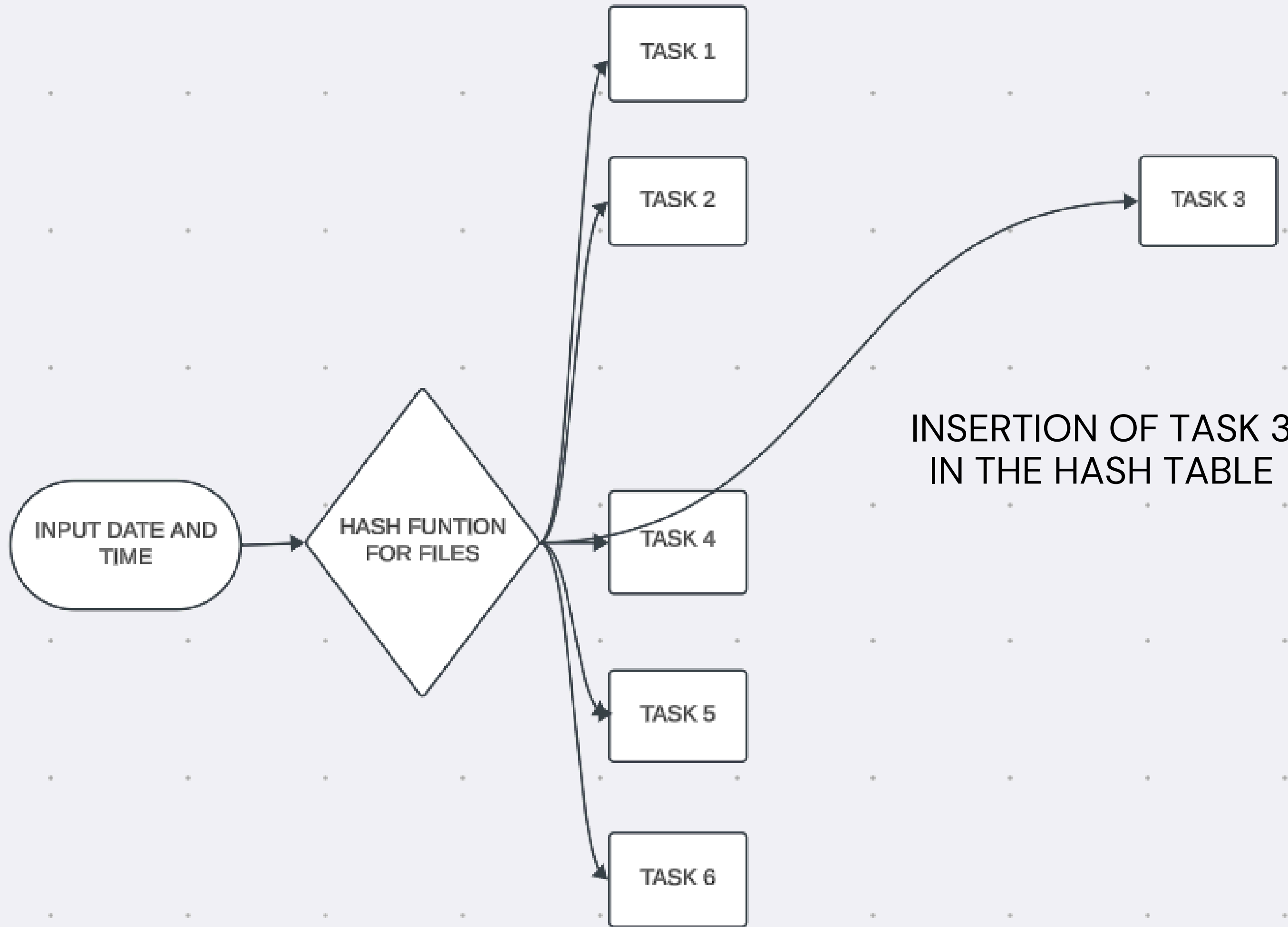


# ALTERNATE ALGORITHM

Insertion time complexity:-  $O(1)$   
Insertion space complexity:-  $O(1)$   
(as the file is created in constant time in the  
system)



(See image for visualisation)





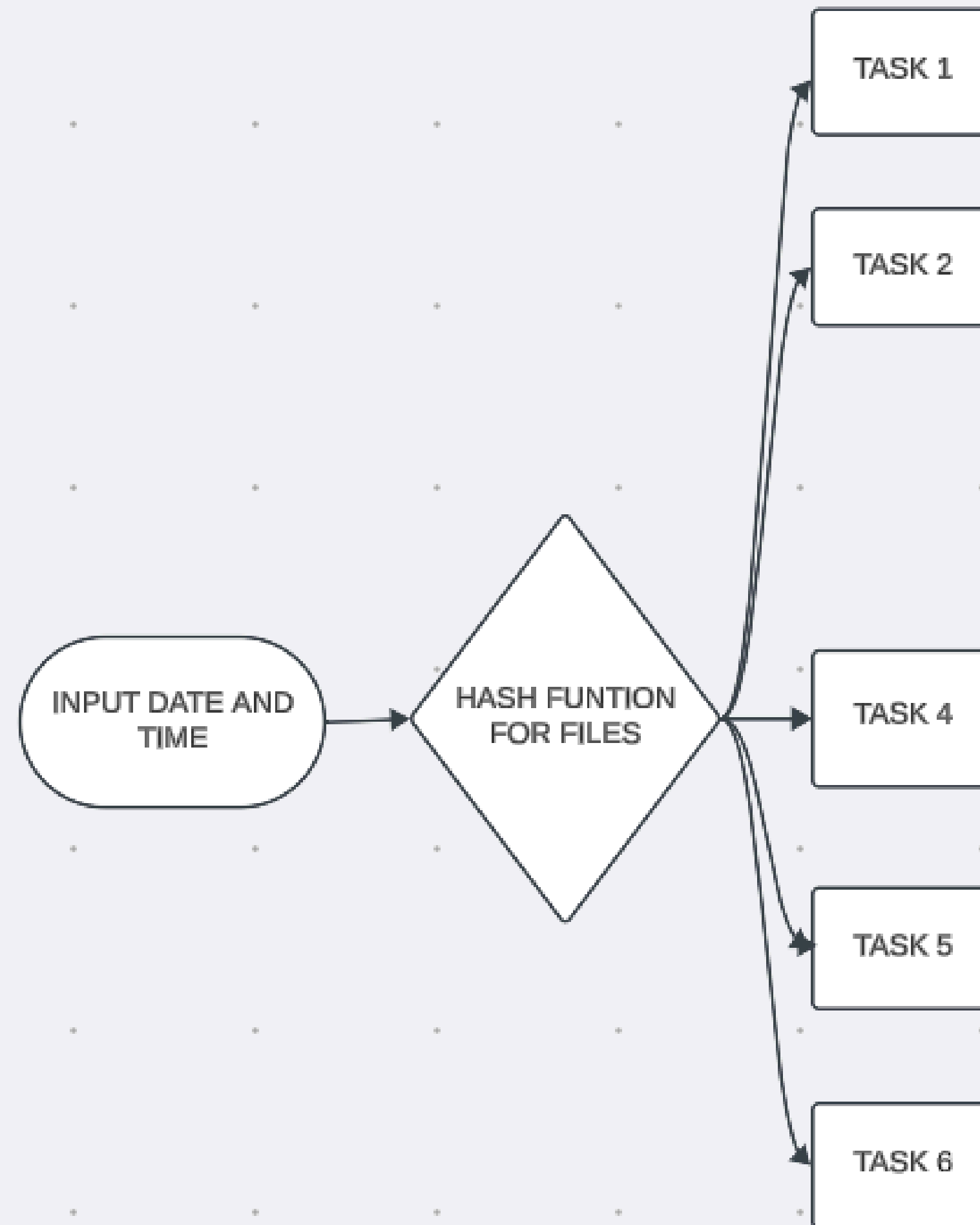
# ALTERNATE ALGORITHM

Deletion time complexity:-  $O(1)$

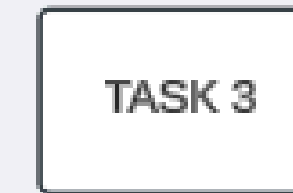
Deletion space complexity:-  $O(1)$

(as the file is deleted in constant time in the system)

See image for visualisation



The access to the pointer  
is released from task 3 ,  
thus it is deleted





# SUMMARY

The insertion and deletion functions are used heavily in the defination of other functions which contriute a major chunk in the implementation of various functionalities.

This were the main algorithm of oursolution and for further detailed information of each function, please refer to the Document attached below.

Link: [Document](#)



# SOME KEY FEATURES WE ADDED:

- Username and password
- Uniqueness of Usernames
- Daily Task Segment
- Editing Task(Every field of task including Rescheduling)
- Maintaining Data even after code is Terminated
- Display of Incomplete Daily Task
- Deletion of Whole tasks list
- Use of Switch cases to increase interaction with the user



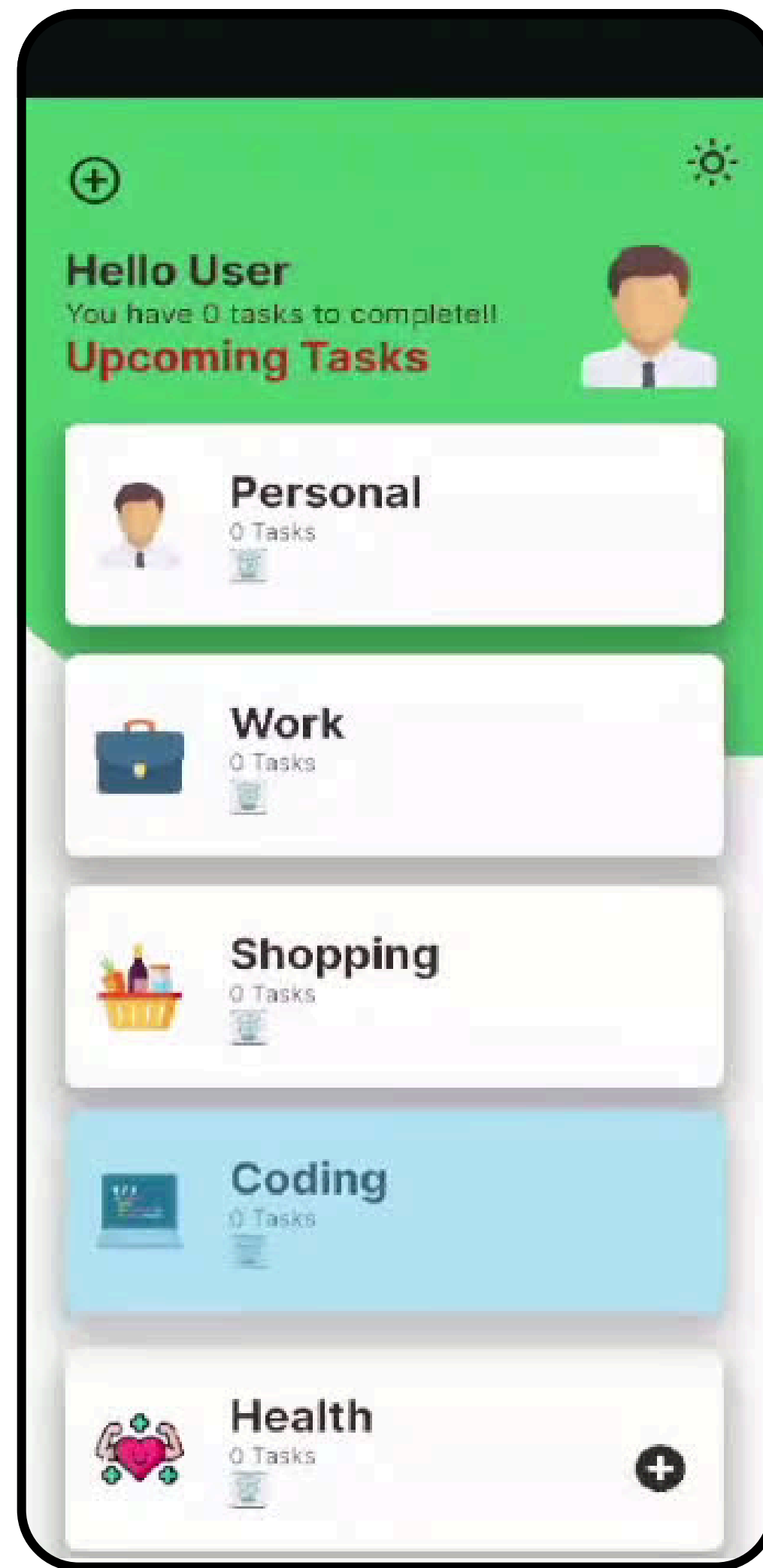
# REGARDING GUI:

We have made a GUI using HTML,CSS and javascript which is made considering mobile usage .

Link To GUI: <https://atikdagu.github.io/todoweb/>







Short Clip of  
our GUI



# VOICE OVER VIDEO

Link: [Voice Over Video](#)



The background features four decorative geometric patterns in the corners. The top-left corner has a series of parallel diagonal lines in a light blue-grey color. The top-right corner contains a cluster of overlapping semi-circles in yellow, red, teal, and dark blue. The bottom-left corner also features a cluster of overlapping semi-circles in red, teal, and dark blue. The bottom-right corner has a series of parallel diagonal lines in a light blue-grey color, mirroring the top-left pattern.

**THANK YOU**