Task management platform

Create a Task management application where you can manage your project by tracking small tasks.

About the platform :

- The application contains multiple **Boards**, to signify **teams**.
- Each board contains different **Lists** to signify **projects.**
- Each list contains different **Cards** to signify **tasks in the projects**.
- Each card can be assigned to a user or may remain unassigned.

Definitions :

- Board: Each board should have an id, name, team members, lists.
- List: Each list should have an id, name and cards .
- Card: Each card should have an id, name, assigned user, priority. Priorities can be P0 (highest), P1 and P2 (lowest).
- User: Each user should have a userId, name, email.

Requirements :

- We should be able to create boards. Boards can be public or private - determining view access across teams as well as delete access across teams.
- Add/remove people from the members list of the Board (team members).
- We should be able to create Lists under a Board.
- We should be able to create Cards under a list.
- Add capability to assign/unassign a member to a Card. Cards should be unassigned by default.
- Capability to delete any Board, List and Card.
    - Deleting a List should delete all cards inside it.
    - Deleting a Board should delete all lists inside it.
- We should be able to view a Board, along with its Lists, and Cards.
- Cards inside a list should be sorted by priority by default - highest(P0) to lowest(P2).
- Build capability to delete User.
- We should also be able to move cards/tasks across lists (lists can be part of other boards/projects). Allow cards/tasks to be moved only to Public privacy boards, unless the member performing this task actually belongs to the board.

Bonus **(only attempt if time permits)** :

- If each Card/task has an attribute called ‘time to complete’, and each person has a ‘time quota’, then only assign a task if ‘time to complete’ <= the person’s remaining ‘time quota’.

Expectations :

1. Make sure that you have working, functionally correct and demonstrable code.
2. Work on the expected output first and then add good-to-have features of your own.
3. Code should be modular, readable and testable.
4. Separation of concern should be addressed.
5. Code should easily accommodate new requirements with minimal changes.

Other Details :

1. Do not use any database or NoSQL store, use in-memory data-structure for simplicity.
2. Do not create any UI for the application.
3. Write a driver class for demo purposes. Which will execute all the features at one place in the code to test all cases.

Sample Test Cases/Data :

BOARD CREATE : should print the id after creation.
“OrdersProjectBoard”, public
“SupplyProjectBoard”, private
“AndroidProjectBoard”, public

USER CREATE : should print the id after creation.
Scott , scott@g.in
Brett , brett@g.in

Tina , tina@g.in
Chan , chan@g.in
Thor , thor@g.in

ADD_MEMBER/REMOVE_MEMBER to a Board : should print whether adding or removing is successful or not.
addMember(scott@g.in, OrdersProjectBoard)
addMember(brett@g.in, SupplyProjectBoard)

BOARD DELETE : should take in Board identifier to be deleted, and User ID of the user who is performing deletion. Print whether deleting is successful or not - return response codes determining reason of failure.
deleteBoard(SupplyProjectBoard, brett@g.in)

LIST CREATE :
createList(OrderProjectList1, OrdersProjectBoard)
createList(OrderProjectList2, OrdersProjectBoard)
createList(SupplyProjectList1, SupplyProjectBoard)
createList(AndroidProjectList1, AndroidProjectBoard)
createList(AndroidProjectList2, AndroidProjectBoard)
createList(AndroidProjectList3, AndroidProjectBoard)

LIST DELETE : Should take in List identifier to be deleted, and User ID of the user who is performing deletion. Should print whether deleting is successful or not - return response codes determining reason of failure.
deleteList(SupplyProjectList1, brett@g.in)

CARD CREATE : Should take in the Card/Task name, List/Project identifier, and the priority of task. This should print the id after creation
createCard(OrderProjectTask1, OrderProjectList1, P0)
createCard(OrderProjectTask2, OrderProjectList1, P2)
createCard(OrderProjectTask3, OrderProjectList2, P1)

createCard(OrderProjectTask4, OrderProjectList2, P1)
createCard(OrderProjectTask5, OrderProjectList1, P0)
…Similarly create data for other Lists.

CARD ASSIGN : Allocates assignee for a task. Should take in the Card/Task identifier, and the User ID to which it is assigned.
assignCard(OrderProjectTask4, scott@g.in)

CARD UNASSIGN : unallocated the assignee for a task.
unassignCard(OrderProjectTask4)

CARD MOVE :
move(OrderProjectTask1, AndroidProjectList) .

CARD DELETE : Takes in the Task/Card identifier and the user ID deleting the Task. Should print appropriate message post deletion.
delete(OrderProjectTask2, scott@g.in)

VIEW BOARD : Takes in the board identifier and the user ID viewing the board. Should print all the lists and cards for the board, cards being in order of priority highest(P0) to lowest(P2).
view(OrdersProjectBoard, scott@g.in)

DELETE A MEMBER
delete(scott@g.in)

MOVE TASK : Takes in the task identifier, board identifiers - source and destination and the user ID doing the operation.
moveTask(OrderProjectTask1, SupplyProjectBoard, OrdersProjectBoard, scott@g.in) - you can choose to pass Lists.