## **Presentation Layer** Service Layer UserService **BoardService** + changePassword(username: string , oldP: string, newP: string): json + createBoard(Name: string, userEmail: string) + addTask(title: String, Description: String, DueDate: String): json + createUser(username: string, password: string): json + login(username: string, password: string): json + nextState(email: string, boardName: string, State: int, TaskID: int): json + getBoard(BoardName: String): json + deleteBoard(Name: string, userEmail: string) + getInProgress(): json + SetColumnLimit(ColName: string, BoardName: string, ColState: int) + logout(userEmail: string):json SetColumnLimit(ColName: string, BoardName: string, ColState: int, NewLimit: int) TaskService + createTask(Title: string, Description: string, DueDate: string): json + EditTitle(newTitle: string, user: User, board: string): json + EditDueDate(newDate: string, user: User, board: string): json + EditDescription(newBody: string, user: User, board: string): json + GetState(string: username, string: BoardName): json **Board** UserController Backlog(Collection) - users: Dictionary<string, User> - InProgress(Collection) - loggedIn: Collection - Done(Collection) - TotalNumOfUsers: int + Name(String) - maxTasks: int = -1 + CreateUser(password: string, email: String): void - numTasks: int + DeleteUser(email: string): void + UserExists(email: string): boolean - Owner(string) + GetUser(email: string): User -BoardID: int + validatePassword(user: User, password: String): boolean + login(email: String, password: String): void + logout(user: User): void + addTask(title: String, Description: String, DueDate: String): void User

## **Changes Made before starting Milestone 2**

**Changes Made after submitting Milestone 1:** 

HasAnyBoard- Check if a BoardController is empty - for exception

GetColList - in order to get all tasks in a given column in a board

GetNameOrdinal - Returns the name of a column given its id in

A User object no longer holds a collection of their boards and

The class user is therefore connected only to the UserConroller,

The checks in order to get a user's board and tasks are preformed

UserController.IsLoggedIn(). This allows a better (lower) duality

Setting the max number of tasks in now done from the Board

it is an attribute of a board rather than a task. It can be said that

GetAllInProgress - See design changes for explanation

IsLegalPassword- in order to validate the password

IsValidEmail- To validate the email address at registration

HasThisBoard- Checks if a BoardController has a board - for

Created:

Created:

IsHebrew - to validate email

exception handling purposes

Added getters and setters in Board

order to fulfill GradingService requirements

handling purposes

Design changes:

which limits our duality.

between the classes.

through calling an instance of

(rather than Task). This is done because

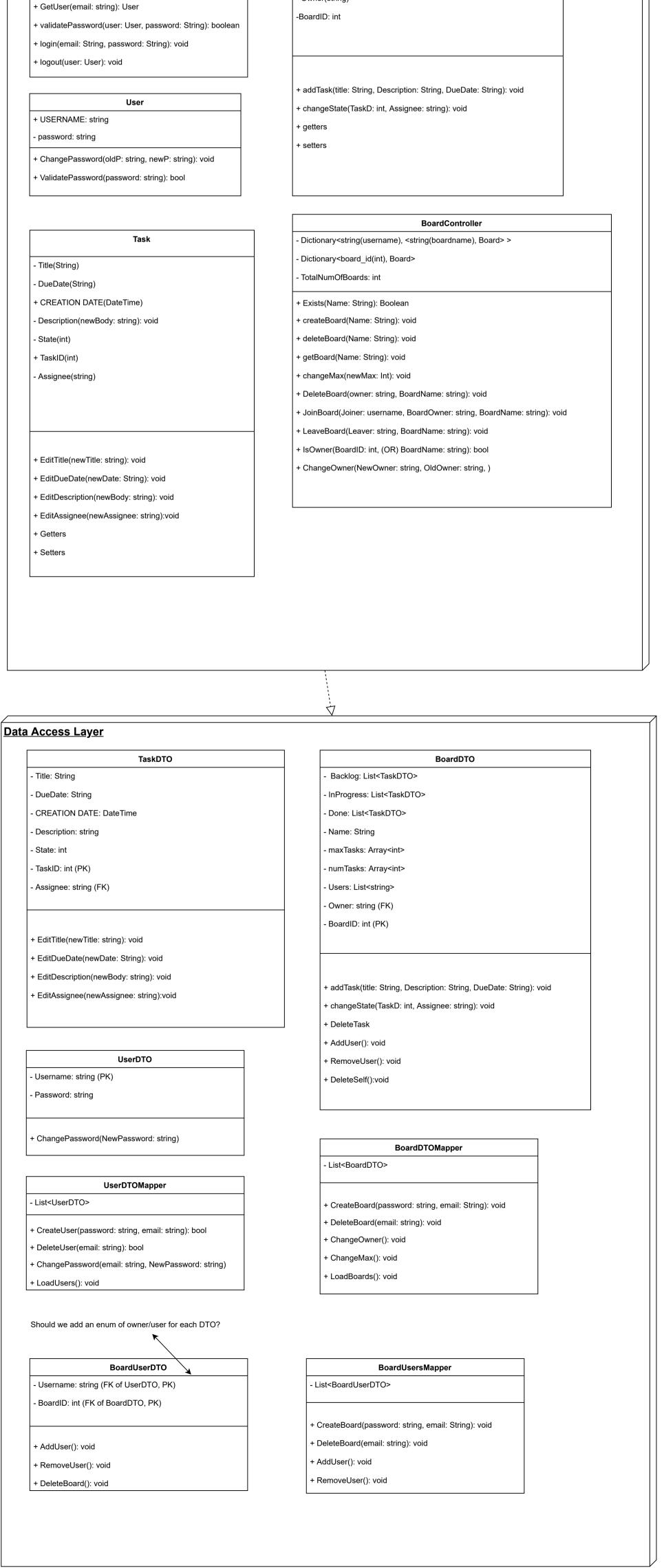
initially we made a design mistake.

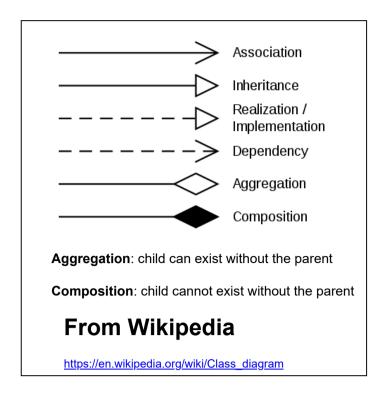
AddTask - same as max tasks.

INew methods for all new functional requirements. Data Access Layer DTO's and Mappers.

Design changes:
None

```
Business Layer
                                                                                + changeState(TaskD: int, Assignee: string): void
       + USERNAME: string
                                                                                + getters
       - password: string
                                                                                + setters
       + ChangePassword(oldP: string, newP: string): void
       + ValidatePassword(password: string): bool
                                                                                                                 BoardController
                                 Task
                                                                                Dictionary<string(username), <string(boardname), Board> >
                                                                                Dictionary<board_id(int), Board>
        - Title(String)
                                                                                - TotalNumOfBoards: int
       - DueDate(String)
       + CREATION DATE(DateTime)
                                                                                + Exists(Name: String): Boolean
       - Description(newBody: string): void
                                                                                + createBoard(Name: String): void
                                                                                + deleteBoard(Name: String): void
       - State(int)
       + TaskID(int)
                                                                                + getBoard(Name: String): void
                                                                                + changeMax(newMax: Int): void
       - Assignee(string)
                                                                                + DeleteBoard(owner: string, BoardName: string): void
                                                                                + JoinBoard(Joiner: username, BoardOwner: string, BoardName: string): void
                                                                                + LeaveBoard(Leaver: string, BoardName: string): void
                                                                                + IsOwner(BoardID: int, (OR) BoardName: string): bool
       + EditTitle(newTitle: string): void
                                                                                + ChangeOwner(NewOwner: string, OldOwner: string, )
       + EditDueDate(newDate: String): void
       + EditDescription(newBody: string): void
       + EditAssignee(newAssignee: string):void
       + Getters
       + Setters
```





**Visibility** + public private

# protected