## Changes in design(2nd submission milestone 1)

added a class named "Authenticator". we had to deal with the problem that the board cade didn't know who are the users. So we added this class in order to have a shared facade didn't know who are the users. So we added this class in order to collection of the logged in users. For that reason, the following classes got an authenticator field added: userService, userFacade, boardFacade.

2. We added to the user facade a method called CheckLoggedin that commi

We added the method inProgressTasks, to answer a requirement we failed to answer in the first UML.

5. We added the method FindBoards in boardFacade, that helps us find a specific board in a dictionary

6. We added the method FindBoard in boardFacade, that helps us find a specific board in a life

7. We added the method FindTask in boardFacade, that helps us find a specific task in a list

8. We added the method Doesl IserExist, that helps us find out whether a user exists or not

## Changes in design(2nd submission milestone 1)

user service + user facade: Load data, delete data.

board service + board facade: DeleteBoard, TransferOwnership, LeaveBoard, JoinBoard

GetLiserBoards LoadData DeleteData

\* the reason we added the methods: loadData, DeleteDate to the user service and the board service, is to maintain low coupling and high cohesion, so that every part of the project will bad and delete seperately.

We also added another layer, the data access layer.

the data access layer consists of controllers, that speak directly to the DB, and Data access objects, or DAO's for short.

each controller, has 4 basic methods and other getters, the 4 methods:

insert: inserts a single object to the DB.

delete: delete a single object from the database

select: retrieves a single object from the database

update: updates a single object.

## Service Layer +Register(email:String,password:String): string + Login(email:String,password:String): string +Logout(email:String): string + LoadData():string + DeleteData():string

BoardService - BF: BoardFacade nard/boardName:String\_email:String): string DeleteBoard (email:String, boardId:int): string

+LimitColumn(email: String, boardName:String, columnOrdinal: int, limit: int): string

AddTask(email: String, boardName:String, title: String, description: String dueDate: dateTime): string +AdvanceTask(email:String , taskld:int, boardName:String, columnOrdinal:String): string

\*GetColumnLimit (email:String ,boardName:String, columnOrdinal:String): string

\*GetColumnName (email:String ,boardName:String, columnOrdinal:int): string

+GetColumn(email:String, boardName:String, columnOrdinal:String): string

+LeaveBoard(email:string, boardID:int): string

+TransferOwnership(CurrentOwnerEmail, newOwnerEmail, BoardName:string): string

\*JoinBoard(email:string, boardID:int ) : string

GetUserBoards(email:string) : string

+ DeleteData():string

TaskService

+UpdateTaskDueDate(email: String ,taskld:int , dueDate: dateTime, boardName:String, columnOrdinal int):string +UpdateTaskDescription(email: String ,taskld int , description: String, boardName: String, columnOrdinal int).string

+UpdateTaskTitle(email: String ,taskld:int , title: String, boardName:String, columnOrdinal: int): string -AssignTask(email: String ,boardName:String,columnOrdinal: int, taskid:int, EmailAssignee : String): st

+InProgressTasks(email:string) : string

+email:String BoardSL

TaskSL

+BoardID : int

eduaDate:dateTime

title:String description:String +assignee:String

Response +ErrorMessage:String

\*ReturnValue:Object

ServiceFactory

BS: BoardService

el nadData(): unid

+DeleteData(): void



